AIAA Modeling and Simulation Technologies Conference

A
Collection
of
Technical
Papers

New Orleans, LA

August 11 - 13, 1997



For permission to copy or republish, contact:
American Institute of Aeronautics and Astronautics
1801 Alexander Bell Drive
Suite 500
Reston, VA 20191-4344

A Collection of
Technical Papers

AIAA Modeling and Simulation Technologies Conference

New Orleans, LA
August 11- 13, 1997

Copyright 1997 © by
American Institute of Aeronautics and Astronautics

ISBN
1-56347-236-8

All rights reserved. No part of this volume may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying and recording, for any purpose without permission from the publisher.

Printed in the U.S.A.

AIAA Modeling and Simulation Technologies Conference and Exhibit

New Orleans, LA / August 11–13, 1997

TABLE OF CONTENTS

Paper No. Title and Author ..... Page No.

Session 11-MST-1 Motion Systems

97-3503 An Initial Evaluation of the Effects of Motion Platform and Drive Characteristics
W. Chung, J. Schroeder, and D. Robinson 1

97-3504 Optimization of Six-Degrees-of-Freedom Motion Systems for Flight Simulators
S. Advani, M. Nahon, N. Haeck, and J. Albronda 12

97-3505 Low Cost Simulation and Motion Cueing
W. Egger and J. Stegner ..... NA

97-3506 Is There an Optimum Motion Cueing Algorithm?
W. Wu and F. Cardullo 23

97-3507 Second-Generation Electric Motion Systems: Advantages and Limitations
R. Verbeek, H. Eynden, and S. Advani ..... WD

Session 12-MST-2 Rotorcraft and Air Cushion Vehicle Dynamics & Modeling

97-3508 Integrating a Computational Fluid Dynamics Ship Airwake Model with a Piloted Simulation
D. Gaublomme, J. Nichols, J. McCrillis, and J. Funk ..... WD

97-3509 Comanche Player Station Comanche Simulation in the Aviation Warfighting Cell
C. Romanoff 30

97-3510 Rotor System Load Monitoring Using a Neural Network Based Approach
K. McCool, L. Flitter, and D. Haas ..... NA

97-3511 Modeling and Simulation for Small Autonomous Helicopter Development
E. Johnson and P. DeBitetto 41

97-3512 Effects of Unsteady Fan Response on Heave Dynamics of an Air Cushion Vehicle Bag and Finger Skirt
J. Chung and P. Sullivan 52

Session 13-MST-3 Pilot Training & Low Cost Simulation

97-3513 Simulation of Fixed-Wing Air-to-Air Combat Using ITEMS
P. Howells, M. Charbonneau, and B. Kwan 62

97-3514 A Low Cost PC-Based Aviation Training Device for IFR Flight Simulation
M. McCabe and J. Sanderson ..... NA

97-3515 Microsoft Flight Simulator as a Data Capture Tool for an Intelligent Tutoring System in Private Pilot Training
R. Hunter and G. Chubb 69

97-3516 Microsoft Flight Simulator Suitability for Cross Country Exercises for Private Pilot Training
G. Chubb and P. Macy 75

97-3517 Virtual Reality Entertainment Technology Applied to Military Parachute Simulation Flight Training
J. Hogue, R. Allen, and R. Smith 83

N/A - Not Available

WD - Withdrawn

Session 14-MST-4 Weapons and Engagement Modeling & Simulation

97-3518 Capability for Analysis of Cruise Missile System Components Using Real Time Simulations
C. Bouchard, J. Broussard, C. Gemmill, R. Hartman, S. Leite, E. Werme, and D. Wright 90
97-3519 Lessons Learned in the Use of Real-Time Simulation for Technology Assessment
B. Stadler and J. Nalepka.....100
97-3520 Modeling the Dynamics of Undersea Vehicles Deployed from Surface/Air Platforms
S. Moyer and K. Watkinson 110
97-3521 Kinetic Kill Vehicle Hardware-in-the-Loop (KHILS) Low Cost Autonomous Acquisition Seeker Hardware-in-the-Loop Testing
L. Byrd III, R. Thompson, and G. Harbaugh.....WD
97-3522 THAAD Integrated System Effectiveness Simulation (TISES)
D. Horn, P. Colvert, and B. Scrip.....121
97-3523 TISES Many-On-Many Engagement Planning Algorithm
J. Asburn, D. Horn, and B. Scrip 497

Session 27-MST-5 Simulator Network & Information Technologies

97-3591 Simulation Meets Information: Bringing Aircraft Modeling and Simulation into the Information Age
W. Beavin.....131
97-3592 An Evolution in Advanced Distributed Simulation
C. Bouwens and A. Zabek.....139
97-3593 Transitioning from DIS to the High Level Architecture (HLA)
D. Fullford 147
97-3594 The Use of Intelligent Software Agents in Distributed Simulation
S. Lubbers, G. Valentino, K. Scribner, and S. Thompson .....WD
97-3815 An Authoritative Data Source for Fixed-Wing Aircraft Kinematics
T. Persing, J. Hammond, and F. Haddix.....WD

Session 28-MST-6 Visual, Radar, and Environmental Modeling & Simulation

97-3595 Overcoming the Limitiations of Today's Image Generators
J. Bleak .....NA
97-3596 Algorithm Optimization for Real-Time Simulation
H. Heaton III and J. DeVilbiss 154
97-3597 Comparison of Scattering Models for Predicting Radar Multipath Effects over the Sea
D. Bucco and J. Chisholm 161

Session 42-MST-8 Test & Evaluation

97-3665 Industry Initiative For Revised Training Simulator Validation Process
K. Neville.....172
97-3666 Validating NASA's "Surface Development and Test Facility" Air Traffic Control Tower Simulator
R. Haines.....181
97-3667 Evaluation of the Navy's F/A-18 A/D Powered Approach Aerodynamics Model
M. Bonner and D. Gingras.....NA
97-3668 Testing SAR Motion Measurement Flight Code in a Unix Environment
T. Kim.....190

NA - Not Available WD - Withdrawn

Session 43-MST-9 Space Systems

97-3669 Space-Based, Ultra Lightweight Imaging Technology Experiment Modeling in the Spacecraft Simulation Toolkit
R. de Jonckheere, B. McGlamery, D. Mergens, and T. Franklin 194
97-3670 Continuous Linear Impulse Balance Method for Real-Time Simulations of Contact Dynamics
F. Belanger and X. Cyril 204
97-3671 High Fidelity GPS Satellite Simulation
M. Baker, B. Dingman, and W. Gregg 213
97-3672 International Space Station Part Task Trainers
D. Forrest, J. Christman, M. Wiederholt, and R. Rodriquez 224
97-3673 Improved Method of Generating Control System Model Using Modal Synthesis
T. Ghosh 231
97-3674 Simulation Technology for Future ESA Space Projects
J. Miro ..... NA
97-3675 PC-Based Distributed Parallel Processing for Real-Time Simulation of Spacecraft Attitude Control
Y.-Y. Lin and S.-S. Liu 240
97-3833 An Overview of Hardware-in-the-Loop Simulation for Missiles
R. Jackson ..... NA

Session 56-MST-11 Simulator Fidelity

97-3728 PC-Based Simulation of the F-16/MATV
M. O'Rourke, J. Ralston, J. Bell, and S. Lash 481
97-3729 Identification of DO-328 Aerodynamic Database for a Level D Flight Simulator
R. Jategaonkar and W. Mönnich 248
97-3730 Concepts for Detecting Pilot-Induced Oscillation Using Manned Simulation
B. Nguyen and T. Cord 259
97-3731 A Comparison of Real-Time Solutions to All-Attitude Angles of an Aircraft
S. Shin and S. Yoon 269

Session 70-MST-13 Aircraft Dynamics, Modeling & Performance

97-3786 Dynamic Inversion Applied to the F-117A
R. Colgren and D. Enns 275
97-3787 Simulation Study of VISTA/F-16 Maneuverability Enhancement Using Forebody Vortex Control
P. McKeehen and T. Cord 285
97-3788 Blended Wing Body Simulation
D. Cameron and D. Biezad ..... NA
97-3789 Fuzzy Logic Based Function Modeling in Aerodynamic Simulations
D. Hensley 296
97-3790 Simulation of a Tailless Aircraft Using Nonlinear Indicial Response Aerodynamic Modeling Methods
P. McKeehen, J. Myatt, D. Grismer, and J. Buffington 305
97-3791 Aeroelastic Modeling of Macrobody Aircraft for Real-Time-Flightsimulation
M. Dahms, W. Kindel, R. Liebich, and O. Schmidt 316
97-3792 Modelling of Aircraft Flight by Means of Dynamically Similar Models with a Flight Control Systems Similarity
S. Sadovnitchii, V. Ryabkov, A. Rushenko, and J. Sandoval 326

NA - Not Available WD - Withdrawn

97-3793 CAD Embedded CAE Tools for Aircraft Designers as Applied to Landing Gear
M. Schmidt and C. Paulson 333

Session 71-MST-14 Simulator Development & Software Reuse

97-3794 The Boeing Commercial Airplane Engineering Simulation, New Airplane Project, Management Observations
G. Williams 341
97-3795 Modern Component Methods for Air Transport Engineering Simulation Software
W. Walker and R. Kircher Jr. 351
97-3796 The Feasibility of Avionics Simulation versus Stimulation in Commercial Aircraft Flight Simulators
K. Conzachi 359
97-3797 SimGraph—A Flight Simulation Data Visualization Workstation
J. Kaplan and P. Kenney 366
97-3798 Application of a Multipurpose Simulation Design
J. Bell and M. O'Rourke 490
97-3799 Model Reuse for Software Development and Testing: The Application of Common Interfaces to Support Variable Fidelity Models
W. Bezdek, T. Halley, and P. Hummel 376
97-3800 Applications of Display Prototyping and Rehosting Tools to the Development of Simulator, Flight Training Device, and Cockpit Displays
P. Bennett 387
97-3814 Composite Materials Technologies Reduce Cost of High-Dollar Components for Maintenance Training
J. Exter 397

Session 72-MST-15 Human Factors

97-3802 Visual-Vestibular Interactions in the Perception and Control of Aircraft Motions by the Pilot
R. Hosman 402
97-3803 A New Approach to Sensory Integration Modeling
G. George and F. Cardullo 412
97-3804 The Human Factors of FMS Usage In The Terminal Area
K. Lee, B. Sanford, and R. Slattery 422
97-3805 Three-Dimensional Versus Two-Dimensional Displays for Air Traffic Control
M. Brown and M. Slater 431

Session 73-MST-16 Research & Test Facilities

97-3806 Dynamic Linking of Hardware-in-the-Loop Simulators at the China Lake Simlab
J. Annos ..... WD
97-3807 Control and Dynamics Simulation Facility at Hughes Space and Communications
S. Zammit 442
97-3808 Mission Equipment Simulation and Evaluation Facility
L. Roumeliotis 449
97-3809 SIMONA—A Reconfigurable and Versatile Research Facility
S. Advani, E. van Zuylen, and S. Bettendorf 458
97-3810 Animation as an Aid to Mishap Investigation and Training
B. Kerr 467
97-3811 Development of ATC Simulation Facility
K. Shiomi, H. Sato, Y. Fukuda, K. Kageyama, T. Hiwada, and K. Shima 471

NA - Not Available WD - Withdrawn

AN INITIAL EVALUATION OF THE EFFECTS OF MOTION PLATFORM AND DRIVE CHARACTERISTICS

William W. Chung*, Jeffery A. Schroeder†
NASA Ames Research Center
Moffett Field, California

Doug J. Robinson††
Lockheed Martin Skunk Works
Palmdale, California

Abstract

Six motion cueing configurations were developed to investigate facility dependent effects on simulation fidelity. These configurations were tested on a large-amplitude motion-based flight simulator and, when possible, on a small amplitude hexapod flight simulator. On the large amplitude device, one configuration had no motion attenuation, and it served as the thruth-case motion reference. Five other configurations were developed to represent a continuum of motion fidelity from high to low. Results from this investigation indicate that only the one-to-one motion configuration consistently reflected the predicted handling qualities ratings based on an existing handling qualities specification. It was also noted that reductions in motion fidelity can falsely improve handling qualities ratings. Non-trivial differences were measured between the facilities, and those differences are currently being investigated.

Nomenclature

g	gravitational acceleration, ft/sec^2
$L_{\delta_{lat}}$	helicopter roll control power, rad/sec^2/in
L_p	helicopter roll acceleration due to roll rate, 1/sec
p	helicopter roll angular rate, rad/sec
p_{cab}	simulator roll rate response, deg/sec
p_{cmd}	simulator roll rate motion command, deg/sec
$\dot{p}$	helicopter roll angular acceleration, rad/sec^2
$\dot{p}_{cab}$	simulator roll angular acceleration response, deg/sec^2
$\dot{p}_{cmd}$	simulator roll angular acceleration command, deg/sec^2
q_{cab}	simulator pitch rate response, deg/sec
r_{cab}	simulator yaw rate response, deg/sec
r_z	vertical displacement between pilot abdomen and the simulator rotational center, positive downward, ft
$\dot{u}_{cab}$	simulator longitudinal acceleration response, x-body axis, ft/sec^2
v	helicopter translational velocity, y-body axis, ft/sec
$\dot{v}$	helicopter translational acceleration, y-body axis, ft/sec^2
$\dot{v}_{cab}$	simulator lateral acceleration response, y-body axis, ft/sec^2
y_{cg}	helicopter lateral c.g. position, ft
δ_{lat}	pilot lateral stick input, in.
ϕ	helicopter roll attitude, rad
ϕ_{cmd}	simulator roll attitude command, deg

Introduction

Following the technology advancement in digital computers, visual image systems, and displays, the expectations in using flight simulation for research and training has grown substantially. With the faster processor, the throughput delays due to digital implementation in the real-time host computer and the visual image system have been reduced significantly. With the increased computational capacity, more accurate aircraft dynamic characteristics and detailed visual scene content can be modeled and presented to the pilots. Motion-based flight simulators add another dimension of realism to flight simulation. The motion cues provide the pilot with the often needed lead information from sensed accelerations to develop control compensation similar to that developed in flight.

At the same time, the understanding of the essence of simulation cueing fidelity is also developing. The performance of each individual simulation cueing device and the fidelity of the integrated simulation cues[1,2] become increasingly important as the mission of the flight simulation demands higher degrees of realism in order to ensure valid test results. Efforts have been made to develop procedures and criteria for the flight simulation community,[3-6] to serve as a standard that many simulator developers are trying to meet, or have met. Reference 3

* Aerospace Engineer. Member AIAA.
† Aerospace Engineer. Senior Member AIAA.
†† Research and Development Engineer. Member AIAA.

Copyright © 1997 by the American Institute of Aeronautics and Astronautics, Inc. All rights reserved.

identifies key elements that determine the quality of motion cues. Reference 4 suggests performance criteria for motion systems. References 5 and 6 suggest criteria that cover all elements in ground-based transport and helicopter training simulators. However, the data to substantiate these criteria are sparse, and the procedures and criteria do not provide guidelines regarding the integrated cueing fidelity. Specifically, the allowable disparities between the visual cues and motion cues are absent. Integrated cueing fidelity is necessary due to the fact that the primary simulation cues, i.e., visual cues and motion cues, are generated on separate devices and mechanisms. The possibility of that the disparities exist between the visual cues and the motion cues is obvious. Hence, it is not surprising that the results generated from different simulation platforms are different and sometimes contradictory.

Two recent studies[7,8] were conducted on the NASA Ames Research Center's Vertical Motion Simulator (VMS) using the VMS's large displacement capability. Simulation cueing synchronization requirements for a sidestep task with a helicopter model were investigated. In both studies, a near-truth fidelity case was developed, i.e., the visual cues and the roll-lateral motion cues were closely matched by using one-to-one motion without attenuating the motion cues. Disparities between the visual cues and the motion cues were then intentionally introduced by varying the throughput delays in both the visual cues and the motion cues, and by varying the magnitude of the roll and lateral motion cues to investigate the cueing interactions between the visual cues and roll-lateral motion cues. Reference 7 found that synchronized visual and motion cues were critical to pilot/vehicle performance and made recommendations on allowable asynchronization. Ref. 8 suggested guidelines on the amount of motion gain required for several levels of fidelity that is consistent with a revised set of guidelines developed originally in Ref. 9.

Since the majority of motion-based flight simulators use the synergistic six-leg hexapod system, it is of great interest to investigate on a limited-motion hexapod the cueing fidelity requirements derived on a non-synergistic motion platform. Because of the short stroke of the actuators and the mechanically coupled drive characteristics, the synergistic system has to attenuate the acceleration quickly, and has nonlinear displacement limits that depend on the current angular and translational displacement. One approach for synergistic systems is to use high pass filters to heavily washout the translational component of accelerations, which also includes those components providing the rotational-to-translational coordination. This leaves the synergistic system to primarily rely on the rotational degrees-of-freedom to provide the needed angular accelerations and specific force motion cues. These washout filters, however, have significant effects on motion fidelity.[9]

The main objectives of this study included the investigation of visual, roll, and lateral cueing fidelity requirements on two different types of motion platforms: synergistic and non-synergistic (independent). A two degree-of-freedom (DOF) helicopter model with a roll rate command, plus a fully coordinated roll-lateral response was implemented on both platforms in order to focus the investigation on the roll and lateral motion fidelity requirements, and to calibrate the roll and lateral motion cues. The experimental differences between the two platforms were minimized by using the same model response, static force feel characteristics, visual scene content, field-of-view, motion washout, and task. Therefore, the results from the two motion platforms could be compared side-by-side such that the effects due to physical differences between the two platforms could be revealed.

Experiment Description

Test Facilities

Two motion-based flight simulators were used for this study. The VMS, as shown in Figure 1, is a six degree-of-freedom (DOF) large-displacement motion platform.[10] Only roll and lateral motion axes were used for this study. The motion capabilities in these two axes are shown in Table 1. The visual system utilized an ESIG-3000 image generator by Evans & Sutherland. The cockpit field-of-view of the R-cab is shown in Figure 2. The visual system had a measured transport delay of 60 msec from the host computer command to a full screen image update. The VMS was fully instrumented for all motion states, which includes angular and translational displacements, rates, and accelerations.

The Lockheed Martin Skunk Works Research & Development Simulator (LMSWRDS) is a six-DOF hexapod motion platform. The LMSWRDS system includes a CAE six degree-of-freedom, six actuator, Model 750 motion system. Its roll and lateral motion capabilities are also shown in Table 1. The simulator includes a visual system capable of very low transport delay (25 milliseconds) measured from host computer output to when the three channel visual display is completely updated. An additional 33 msec was added to the visual command to make a total visual throughput delay of 58 msec to closely match that used in the VMS. The cockpit has a field-of-view of 139 degrees of azimuth and 26 degrees of instantaneous elevation which is similar to the VMS's FOV with the exception of the VMS's extra chin-window. This difference, however, was estimated by the pilots to be insignificant due to the nature of the task, which required their main attention on the center window. A four axis accelerometer package (roll, longitudinal, lateral, and heave in body-axis) and a three-axis angular rate gyro package were installed in the motion compartment near the pilot's headrest. The motion system response and other simulation model response data were displayed on strip-chart recorders and collected digitally for data analysis.

Aircraft Model, Force Characteristics, and The Task

A two DOF helicopter model with a roll rate command and a

fully coordinated roll-lateral response about hover is given by equation 1,

$$\begin{bmatrix} \dot{p} \\ \dot{v} \\ \dot{\phi} \end{bmatrix} = \begin{bmatrix} L_p & 0 & 0 \\ 0 & 0 & g \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} p \\ v \\ \phi \end{bmatrix} + \begin{bmatrix} L_{\delta_{lat}} \\ 0 \\ 0 \end{bmatrix} \delta_{lat} \qquad (1)$$

The same model was implemented at both VMS and LMSWRDS with the roll accelerations due to roll rate (or roll damping stability derivative), L_p, set at -4.5 1/sec and the roll control power, $L_{\delta_{lat}}$, at 0.4 rad/sec^2/in. The digital host computer's frame time was 10 msec in the VMS and 33 msec in the LMSWRDS.

The lateral stick force feel characteristics at the LMSWRDS were used as the reference for both facilities and are shown in Figure 3. The force feel gradient at the VMS was set at ±2.5 lb/in with a 1 lb breakout force. The physical geometry of the sticks in the two facilities, however, was different. The VMS had a conventional center stick with the lateral pivot point located 26 inches below the hand-grip reference point. The center stick in the LMSWRDS had a pivot point that was at 11 inches below the same reference point. Therefore, the static force-to-roll rate response was the same between the two facilities. The roll stick angular displacement, however, would be greater at the LMSWRDS than the VMS. Two other force gradient characteristics were simulated in the VMS: 4.4 lb/in with 1 lb of breakout force and 0.6 lb/in with 0.5 lb of breakout force, to check the handling qualities dependency on the stick force characteristics. According to the *Handling Qualities Requirements for Military Rotorcraft*[11], the 0.6 lb/in force gradient was just above the lower bound of the specified allowable Level 1 force gradient. The 2.5 lb/in force gradient was just below the upper bound of Level 1, and the 4.4 lb/in fell within the Level 2 boundaries. A 0.6 lb/in force gradient was used in Reference 7 and 8. In all cases, $L_{\delta_{lat}}$ was adjusted to maintain a constant roll-rate to stick-force response as the referenced force-feel configuration.

The task was a 20 ft sidestep task performed at a constant altitude of 25 ft as shown in Figure 4. The piloted task started with a 20 ft translation to the right followed by 10 seconds of hovering at the right stationkeeping point. This was followed by a 20 ft translation to the left followed by another 10 sec hover at the left stationkeeping point. Each stationkeeping point was denoted by the center of the two red circles, and a noseboom was placed on the aircraft to allow for alignment with the circle's center. The black and white striped visual layout with two designated red hover circles were produced at both facilities with the same dimensions. Pilots were instructed to complete the right and left sidesteps in one smooth maneuver within a specified time. The desired time to complete the lateral translation was 6 seconds. The adequate completion time was 10 seconds. The desired stationkeeping position error was ±2 ft. The adequate position error was ±5 ft. Three experienced pilots participated in the light (0.6 lb/in) and heavy stick (4.4 lb/in) experiments and 4 pilots participated the medium stick (2.5 lb/in) experiment on the VMS. Pilots were asked to give handling qualities ratings (HQRs)[12] and motion fidelity scale (MFS) ratings as shown in Table 2.

Motion Cues Response

The motion system responses in both facilities were checked using the frequency response technique developed for system identification called CIFER®[13]. Two methods were developed to generate adequate input and output power spectra to identify the frequency responses across the frequency spectrum of interest. A white noise with a Gaussian distribution, Figure 5, was used as the pilot control input in the VMS, and a pilot generated frequency sweep, Figure 6, was used in the LMSWRDS. The longer time history from the white noise generated more lower frequency response and better coherence, which is a linear system characteristic measurement as described by Reference 13, than the pilot generated frequency sweep. Several pilot generated frequency sweep time histories, however, can be concatenated by CIFER® to improve the lower frequency response range and coherence. Both VMS and LMSWRDS exhibited good coherence up to 20 rad/sec, which was beyond the frequency range of the pilot input for the task. The VMS acceleration response-to-command frequency response of the roll and lateral motion are shown in Figure 7, also shown is the guideline suggested by Reference 6. Both VMS roll and lateral motion responses showed an equivalent time delay of 60 msec which matched well with the visual throughput delay. The frequency response of LMSWRDS lateral motion is also shown in Figure 7. The roll response of the LMSWRDS, however, was found to be questionable and is being investigated.

Test Configurations

The same motion cueing software configurations were developed for both simulators so the results could be compared side-by-side. Six motion test configurations were developed: one-to-one (VMS only), high motion fidelity, medium motion fidelity, low motion fidelity, lateral motion, and fixed-based. The one-to-one configuration, which had closely matched visual cues and motion cues without attenuated motion, served as the reference for pilot/vehicle performance and pilot workload. Four of the five test configurations were developed specifically for LMSWRDS (but still tested at both facilities) due to the hexapod's limited displacements. The high and medium motion fidelity configurations were developed based on the magnitude and phase criteria suggested in Reference 9. The low fidelity configuration is the nominally installed configuration at the LMSWRDS. A block diagram that illustrates the mechanization of these three washout configurations is shown in Figure 8. The magnitude and phase responses of the motion-system-acceleration-command

at a sine wave input of 1 rad/sec for these three configurations are shown plotted on the Ref. 9 criterion in Figure 9. The lateral motion commands in all three washout configurations were not attenuated in order to generate the fully coordinated roll-lateral motion cues. Most hexapods are in the low-fidelity region based upon the Ref. 9 criteria, as they must be able to avoid their displacement limits for tasks that have substantially more maneuvering than the limited displacement sidestep task described here. It is because of this limited displacement and maneuvering task that it was possible to develop the medium and high fidelity configurations as shown in Figure 9.

A lateral-only configuration was developed as a result of pilots commenting that they often sensed roll motion while flying fixed-based. The theory to be investigated here was that perhaps a small lateral bump, when combined with the strong roll cues that are perceived from a visual scene, could fool a pilot into believing that both roll and lateral motion were present. This small lateral bump would essentially reduce the onset of vection, thus making a pilot believe he or she is moving, just from the visual cues alone (after the bump). A very small amount of lateral motion was supplied for this bump, since the tradeoff is that an incorrect sideforce cue is provided, when no sideforce cues should be provided from this coordinated model. A similar concept was found useful in a yaw motion cueing study.[14]

The motion drive commands in both facilities were configured such that the lateral motion was fully coordinated and that the roll center of rotation was at the pilot's abdomen, i.e. the lateral specific force at that point should always remain to be zero.

Preliminary Results

The average HQRs from the VMS with the three different force gradient characteristics (light, medium, and heavy) are shown in Figure 10. It is noticed that the motion configuration had effects regardless of the stick gradient. Only in the one-to-one motion configuration, do the average HQRs correctly reflect the expected handling qualities based on the lateral stick force gradient characteristics (and vehicle bandwidth). The HQR results for the light stick in the one-to-one motion configuration were found to be consistent with the results from References 7 and 8. This suggests that one-to-one is a near truth-case motion configuration. For the 2.5 lb/in force gradient, the same correct expectation is only found additionally for the high motion fidelity case. Unfortunately, the 2.5 lb/in stick data for the low motion fidelity and lateral only configurations were invalid due to a setup error.

The average motion fidelity scale ratings are shown in Figure 11, which shows that the medium and low fidelity washout configurations had good correlation with the magnitude and phase criteria. But the high fidelity configuration received averaged medium fidelity ratings from pilots for all three stick force gradient variations. The results suggest that there are other factors influencing the motion fidelity. One of the factors could be attributed to the noise from the VMS lateral track. As the one-to-one and high fidelity motion configurations commanded larger travel, the noise (which was proportional to the lateral-track velocity) became noticeably different from that of the visual flight.

An important, and often neglected, point is illustrated by the 4.4 lb/in data. In the one-to-one configuration, pilots rated the configuration Level 2, and the common complaint was that the vehicle was too sensitive. This same sensitive vehicle was rated better as the roll motion gain was reduced from 1.0 to 0.4 to 0.25. Effectively, the simulator was now giving the pilot a feeling of reduced vehicle sensitivity, as the roll motion acceleration response was reduced (even though the vehicle attitude response shown in the visual scene was unchanged). The HQR became worse again for the low fidelity configuration, most likely because the phase response of the high-pass filter caused miscues. However, the interesting point is that as the motion fidelity decreases, the handling qualities artificially improved, which is the wrong answer here. This problem of an improper simulator evaluation of a vehicle's sensitivity has often occurred, sometimes only reappearing in first flight. Thus, simulator users should attempt to find tasks that allow the true model sensitivity to be evaluated. For small displacement simulators, this typically necessitates a high-frequency task so that the motion gains can be increased while keeping the simulator within its displacement constraints.

The task was flown in the LMSWRDS for a preliminary assessment on the overall simulation fidelity. Pilot comments on the field-of-view and the quality of the visual content were satisfactory. However, pilots felt that coordinated motion was not achieved, which is corroborated by acceleration measurements. Pilots frequently felt a lurching or an out-of-phase sideforce during the stationkeeping part of the task. The recorded lateral accelerometer response confirmed pilots' comments. Those sensations might be consistent with the concerns about the roll motion response discussed earlier. In the high fidelity configuration, pilots actually felt pitch, yaw, and surge when motion commands were within the single-axis limits (see Figure 12.)

Concluding Remarks

1) Motion cues have key effects when used in ground-based handling qualities evaluations. A one-to-one motion configuration was shown to provide a near-truth flight simulation case. The disparities between the visual cues and motion cues when motion attenuation was present had various degrees of influence on the handling qualities, which made the results less predictable.

2) Reductions in simulation fidelity can falsely improve handling qualities ratings, especially when it comes to

evaluating vehicle sensitivity. Simulator users should strive to jointly develop a task that can be evaluated with a vehicle model at near one-to-one motion to reveal potential sensitivity problems.

3) An end-to-end calibration procedure is critical to a research and development oriented flight simulation facility. The procedure must include both time domain and frequency domain verification and validation. The procedure also must include measurements of the major simulation cues, i.e., visual cues and motion cues. Motion cues must be calibrated with a standard set of steps and maneuvers using the washout filters to ensure that no anomalous acceleration or specific force is generated.

4) A follow-on test in the hexapod facility will be conducted once the anomalies are rectified.

References

[1]Mitchell, D. G.; and Hart, D. C.: "Effects of Simulator Motion and Visual Characteristics on Rotorcraft Handling Qualities Evaluations," American Helicopter Society Conference on Piloting Vertical Flight Aircraft, Jan. 1993.

[2]Bray, R.S., "Visual and Motion Cueing in Helicopter Simulation," NASA TM 86818, Sept. 1985.

[3]Working Group: "Dynamic Characteristics of Flight Simulator Motion Systems," AGARD Advisory Report No. 144, AGARD, NATO, Neuilly sur Seine, France, September 1979.

[4]Military Standard MIL-STD-1558, Six-Degree-of-Freedom Motion System Requirements for Aircrewmember Training Simulators, February 1974.

[5]AC-120-40B, Airplane Simulator Qualification, U.S. Department of Transportation, Federal Aviation Administration, July 1991.

[6]AC-120-63, Helicopter Simulator Qualification, U.S. Department of Transportation, Federal Aviation Administration, October 1994.

[7]Chung, W.Y. and Schroeder, J.A.: "Visual and Roll-Lateral Motion Cueing Synchronization Requirements for Motion-Based Flight Simulations," American Helicopter Society's 53rd Annual Forum, April 1997.

[8]Schroeder, J.A. and Chung, W.Y.: "Effects of Roll and Lateral Flight Simulation Motion Gains on a Sidestep Task," American Helicopter Society's 53rd Annual Forum, April 1997.

[9]Sinacori, J. B.: "The Determination of Some Requirements for a Helicopter Flight Research Simulation Facility," Contractor Report STI-TR-1097-1, September 1977.

[10]Danek, G. L.: "Vertical Motion Simulator Familiarization Guide," NASA TM-103923, May 1993.

[11]Aeronautical Design Standard, Handling Qualities Requirements for Military Rotorcraft, ADS-33D, July 1994.

[12]Cooper, G. E.; and Harper, R. P., Jr.: "The Use of Pilot Rating in the Evaluation of Aircraft Handling Qualities," NASA TN D-5153, Apr. 1969.

[13]Tischler, M. B., Cauffman, M.G.: "Frequency-Response Method for Rotorcraft System Identification: Flight Applications to BO-105 Coupled Rotor/Fuselage Dynamics," Journal of the American Helicopter Society, Vol 37, No 3, pgs 3-17, July 1992.

[14]Schroeder, J.A.; and Johnson, W.: "Vertical Axis Rotational Motion Cues In Hovering Flight Simulation," AIAA Flight Simulation Technologies Conference, AIAA 94-3432, August 1994.

Table 1. Roll and lateral motion system limits for the VMS and LMSWRDS

Axis	Limits	VMS	LMSWRDS
Roll	Displacement (deg)	±18.0	±26.0*
	Rates (deg/sec)	±40.0	±25.0*
	Acceleration (deg/sec^2)	±115.0	±160.0*
Lateral	Displacement (ft)	±20.0	±10*
	Velocity (ft/sec)	±8.0	±2.67*
	Acceleration (ft/sec^2)	±16.0	±19.0*

* Maximum limit in a single axis excursion

Table 2. Motion fidelity scale

	Description	Score
High Fidelity	Motion sensations are not noticeably different from those of visual flight	1
Medium Fidelity	Motion sensations are noticeably different from those of visual flight, but not objectionable	2
Low Fidelity	Motion sensations are noticeably different from those of visual flight and objectionable	3

VMS Nominal operational motion limits			
Axis	Displ	Velocity	Accel
Vertical	± 30	16	24
Lateral	± 20	8	16
Longitudinal	± 4	4	10
Roll	± 18	40	115
Pitch	± 18	40	115
Yaw	± 24	46	115
All numbers, units ft, deg, sec			

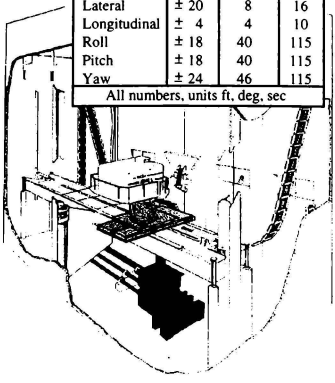


Figure 1. NASA Ames Research Center Vertical Motion Simulator (VMS)

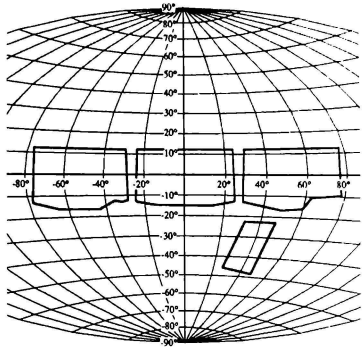


Figure 2. VMS cockpit field-of-view

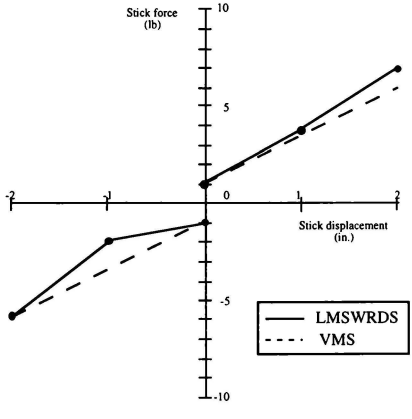


Figure 3. The lateral sitck force gradient characteristics for the Lockheed Martin Skunk Works Research & Development Simulator (LMSWRDS) and the VMS

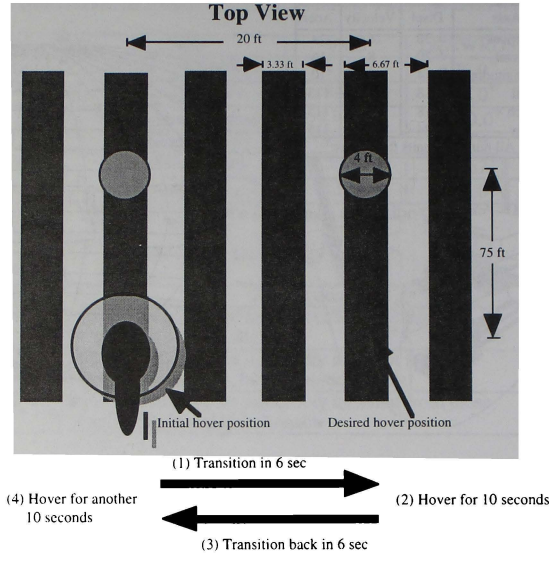


Figure 4. Top view of the task layout

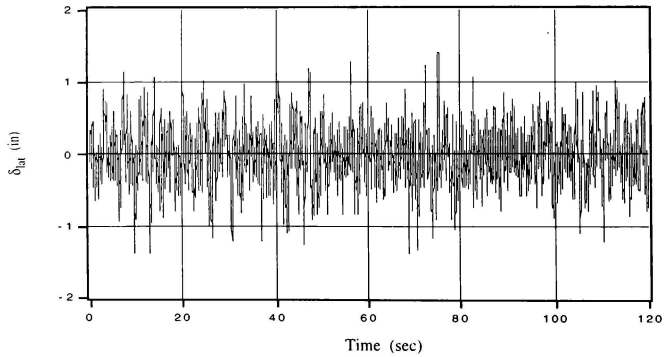


Figure 5. A representative white noise generated lateral stick input

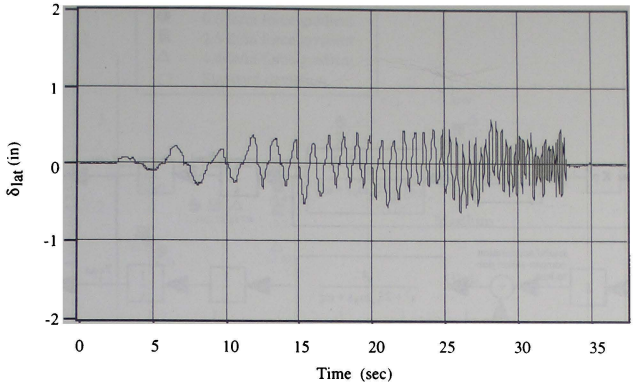


Figure 6. A representative pilot generated lateral stick sweep for system identification

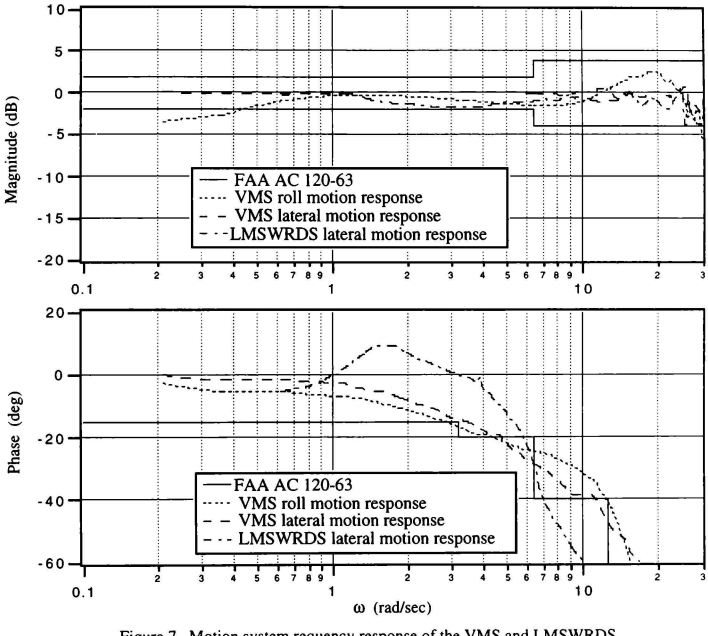
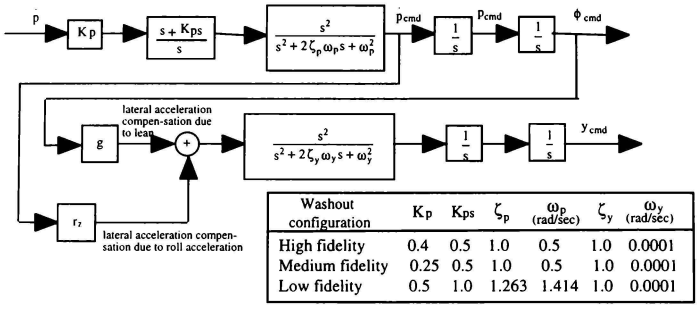


Figure 7. Motion system requency response of the VMS and LMSWRDS



Washout configuration	Kp	Kps	ζ_p	ω_p (rad/sec)	ζ_y	ω_y (rad/sec)
High fidelity	0.4	0.5	1.0	0.5	1.0	0.0001
Medium fidelity	0.25	0.5	1.0	0.5	1.0	0.0001
Low fidelity	0.5	1.0	1.263	1.414	1.0	0.0001

Figure 8. Motion drive command blockdiagram for both simulators

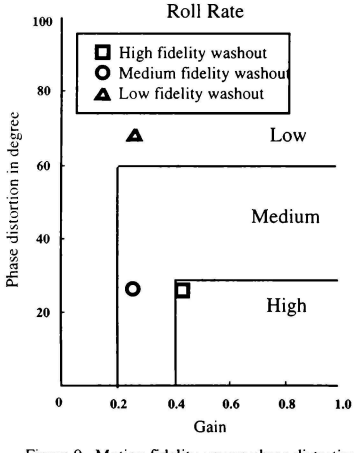


Figure 9. Motion fidelity versus phase distortion and gain at 1 rad/sec for the three washout motion configurations

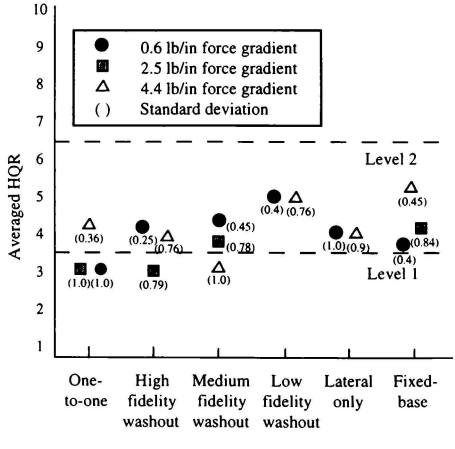


Figure 10. Averaged HQRs with the light, medium, and heavy force gradient in six motion configurations

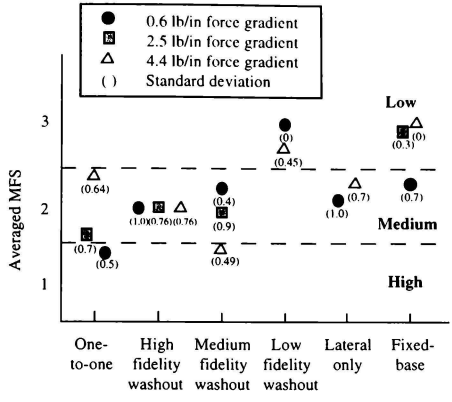


Figure 11. Averaged Motion Fidelity Scale (MFS) with the light, medium, and heavy force gradient in six motion configurations

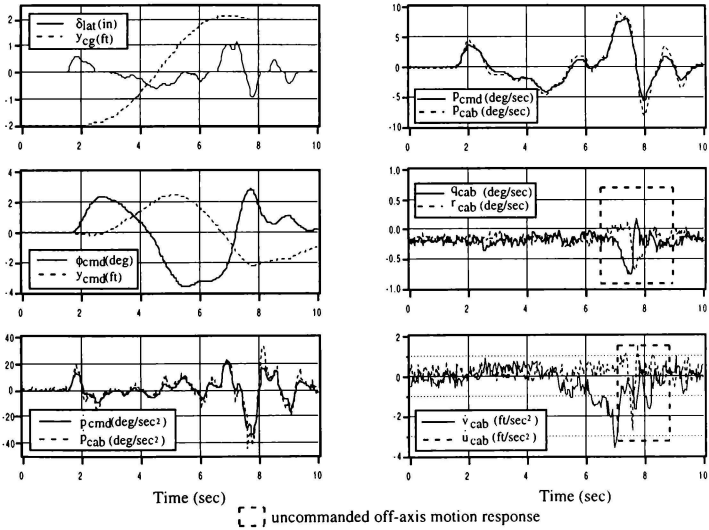


Figure 12 LMSWRDS motion response in a sidestep task

AIAA-97-3504

Optimization of Six-Degrees-of-Freedom Motion Systems for Flight Simulators

S.K. Advani[1], M. A. Nahon[2], N. Haeck[3], J. Albronda[3]

International Centre for Research in Simulation, Motion and Navigation Technologies SIMONA,
Delft University of Technology, Anthony Fokkerweg 1, 2629 HS, Delft, The Netherlands, s.advani@lr.tudelft.nl

Abstract

The cueing capabilities of a synergistic flight simulator motion system are limited primarily by the maximum translational and rotational travel allowed by the motion-base. This travel capability, also known as the workspace, is dictated by the kinematic layout of the motion system. Furthermore, the Jacobian matrix, which maps velocities from platform space to joint space, indicates the dexterity of the mechanism, or the mechanical effort needed by the actuators to move the platform. In order to systematically design unconventional motion-bases, a software program has been developed to analyze arbitrary six-degrees-of-freedom motion systems. This software was then mated to an optimization program, to determine the optimal layout of the motion system, given the workspace performance objectives and the design constraints. This package allows the investigation of unconventional platform geometries and actuator attachment points, thus allowing the designer to tailor the workspace as required by the simulation task, to ensure that a satisfactory dexterity is maintained, and to guarantee that the actuator legs do not interfere mechanically. This paper describes the program, and shows examples of its applications, first to generic workspaces, and then to the workspace required for the simulation of a large transport aircraft.

Introduction

The motion bases of modern flight simulators are generally based on a mechanism known as the Stewart Platform[1] (originally proposed in 1938 for the testing of tires) which is comprised of a base-frame, six prismatic actuator legs (the jacks), and an upper moving platform which carries the payload. The legs are attached in pairs, via gimbal joints, to the upper and lower platforms near the vertices of their triangular frames. Figure 1 shows the general arrangement of a Stewart Platform .

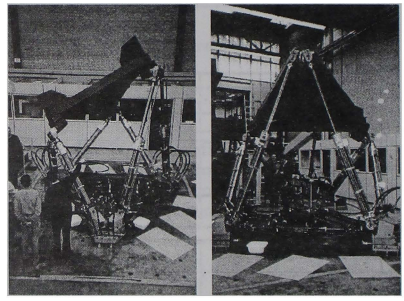


Figure 1 - A modern Stewart Platform [2], comprised of six identical actuators attached to a triangular base-frame and moving platform.

In the conventional arrangement, the locations of the six upper and six lower gimbal joints can be mapped on circles, and the gimbal pairs are separated by a fixed distance (Figure 2). The motion of the legs, all six of which are identical, is constrained by their minimum and maximum lengths. These constraints impose on the mechanism a kinematic envelope which the upper platform can achieve with respect to the inertial reference frame. The envelope of the total kinematic excursion (or travel), known as the workspace, determines the cueing ability of the simulator. Larger workspace in a given direction generally allows longer cue durations. Increasing proportionally the size of *all* of the members of the Stewart Platform simultaneously extends the maximum

1 Director of SIMONA, Assistant Professor, Member AIAA

2 Associate Professor, on sabbatical from U. of Victoria, Canada, Senior Member AIAA

3 Undergraduate student

Copyright © 1997 by the American Institute of Aeronautics and Astronautics, Inc. All rights reserved.

translational capabilities of the upper platform, yet will not affect the rotational limits.

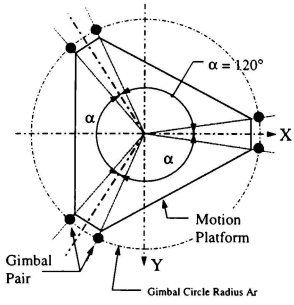


Figure 2 - Typical layout of base-frame or moving platform of a conventional Stewart Platform mechanism.

Freely changing the layout itself, thereby deviating from the standard circular arrangement, is possible. However the designer must prevent the platform from achieving a pose which is at, or close to *singularities*. In this situation, the ratio of actuator displacements to the resulting platform displacements is very low, meaning that the positioning accuracy may suffer, the mechanical loads can become very high, or the control difficult.

Goal of this research

Currently, the design of motion-based flight simulators is carried out by specifying the performance required by the motion cueing mechanism, in order to generate translational and angular motions. These motions are intended to approximate the specific forces and angular accelerations encountered by the pilot in the simulated aircraft, and work in conjunction with the visual cues to generate the perception of self-motion. The Stewart Platform motion-base is sized as a function of these requirements, and other design- and manufacturing-based constraints. Once the simulator is available, a motion drive algorithm is applied to generate the motion cues which would most closely represent those which the aircraft pilot would experience. This motion drive algorithm reads the outputs of the simulated aircraft model, and generates jack length commands for the motion cueing mechanism.[3,4,5] The motion drive algorithm can be heuristically adjusted such that the motion platform will never exceed the limits of the actuators, and will simultaneously provide reasonable inertial motion cues to the pilot.[6] If the kinematic limits are reached however, the pilot may detect a bump in the motion, known as a false cue. On the other hand, if the lower-amplitude motion cues are attenuated, there is a risk that they will not be detected by the pilot. Clearly, choosing appropriate motion is an optimization problem in itself.

If, however, the *spatial* requirements of the motion platform in its *motion-cueing* role, were better known *prior* to its construction, then the architecture of that mechanism could be specified on the basis of these workspace requirements. Rather than trying later to tune a particular motion base, the process could be reversed by tailoring the mechanism to provide the required motion cueing workspace.

The design of the cueing mechanism is by no means restricted to the typically symmetric shape of the Stewart Platform. In fact, the designer can choose in three-dimensional space the locations of each upper and lower leg attachment point, as well as the properties of each leg, all of which influence the resulting workspace, and the usefulness of the motion base. Although there has been some limited investigation of unconventional 6-dof geometries,[7] little prior work exists on the *optimal* design of motion cueing mechanisms.

This paper describes an approach to the optimal design of asymmetric Stewart-type motion cueing mechanisms. This approach is tested using aircraft responses generated from a flight simulation model of a Boeing 747-400, which are fed through a unity-gain classical washout filter[8] to predict the simulator trajectories. Based on these trajectories, a workspace is specified, and a mechanism is designed to best fit that workspace.

Simulator workspace requirements

The factors which effectively determine the simulator workspace requirements are given by the shaded boxes in Figure 3. The workspace *required* for a particular simulation application depends on the vehicle properties, as well as the maneuvers which must be simulated. Although the motion requirements could be generated using off-line simulations, a more realistic approach was adopted here. A full-flight simulator with a mathematical model representing the Boeing 747-400 was flown by a qualified pilot. Thirty-one training-critical maneuvers were recorded. The time histories of the aircraft specific forces and angular rates, were then

passed through a classical motion-drive algorithm.[9] This particular motion-drive law was selected since it does not distort the simulator trajectory through the use of adaptive filters. The scaling factors of the washout algorithm, which had previously been tuned to a similar aircraft type, were made unity, and the input signal limiting was removed.

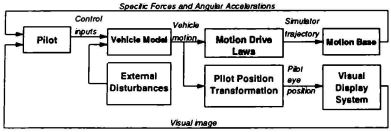


Figure 3 - The process of motion cueing in flight simulators inherently involves calculating the desired motion-base trajectories. Simulator trajectories command the actuators.

Figure 4 shows typical time histories of the resulting simulator centroid motion, for the sway-heave, sway-yaw, sway-pitch and sway-roll motions.

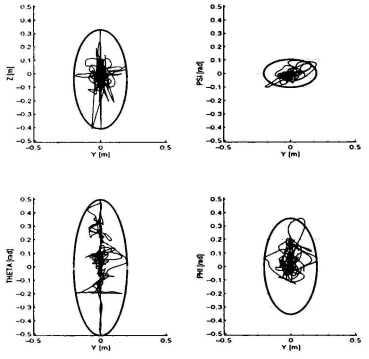


Figure 4 - Trajectory maps of the predicted simulator motions plotted for each pair of degrees-of-freedom (15 combinations in total), represent the measured vehicle model responses to 31 maneuvers, passed through the motion-drive laws. Weighting ellipses indicate mechanism minimum workspace criteria.

The ellipses in Figure 4 circumscribe the maximum excursions in each degree-of-freedom, and will serve as the weighting factors for the design synthesis phase. Due to the six required degrees-of-freedom, fifteen two-dimensional cross-section mappings are required. Finally, all of these are combined to create a "hyper-ellipsoid" in six-space.

Design Freedom

In this work, deviations from the conventional Stewart Platform are introduced in the mechanism design process and the geometry is allowed increasing freedom. In its fully general form, the geometric design of a six degrees-of-freedom synergistic platform consists of determining the six upper gimbal attachment points and the six lower gimbal attachment point in 3-D space, as well as the minimum and the maximum actuator lengths for each of the six actuators. This leaves the designer with 6*3 + 6*3 + 6*2 = 48 free design variables to select. Varying any one of these variables will influence the final performance of the design. Furthermore, the influence of each individual variable on the final performance is nonlinear and dependent on the values of all other variables. Clearly, considering the large number of design requirements as well, this design problem must be addressed in a systematic way if there is to be any hope of solving it.

For the purposes of the present work, the solution of the fully general design was considered excessively complex and was found to be beyond the capacity of our available computing platforms. Thus, the number of design variables had to be reduced to a more realistic subset. After some testing and iteration, the subset chosen was one which resulted in 9 design variables. The mechanisms investigated were symmetrical about the X-Z plane. The location of the upper and lower attachment points on their respective platforms was generalized from a circle (in the conventional Stewart platform) to an ellipse, as shown in Figure 5. The size and aspect ratio of these ellipses were made variable, thus resulting in two design variables per platform. The separation between each pair of attachment points was fixed at the minimum physically achievable (2d for the upper platform and 2p for the lower platform). One pair of attachment points was fixed to lie symmetrically on the x-axis. The other 2 pairs of attachment points were located symmetrically an angle α from the x-axis (and correspondingly an angle β on the lower platform). The angles α and β constituted 2 additional design variables. Finally, the actuator minimum lengths Q_{min} were also used as design variables, but the symmetry of the platforms reduced these to only 3 additional design variables. The actuator maximum lengths Q_{max} were then based on the minimum lengths according to,

$$Q_{max} = 2 \cdot Q_{min} - 0.981 \qquad (1)$$

This represents a family of actuators with identical mechanical hardware, except that the cylinders and pistons are cut to differing lengths. Equation (1) is determined empirically and is based on existing motion base hardware.

The geometry specified by these 9 design variables is substantially more general than that allowed by the conventional Stewart platform and resulted in problems that were solvable in a few hours on our available computing platforms. It was also felt that this reduced set of variables would constitute a good first attempt at the optimal design of a more general class of flight simulator motion base. It should be noted that there is no fundamental reason preventing the generalization of this problem to the fully general one of 48 design variables, though it would be expected that each optimization would then require days or weeks of computing time. In addition, it could be questioned how *practically* viable a fully general configuration might be due to the manufacturing complexities it might entrain.

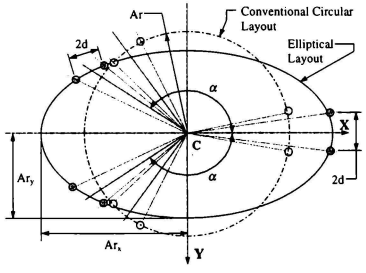


Figure 5 - Layout of standard (circular) and elliptical (shaded) upper platform. Gimbals are mapped along the boundary of the ellipse (rather than circle), and angle α can range between 90° and 170°. Distance 2d is held constant. Similar variations are allowed to the base-frame layout during the optimization.

Optimization Method

To be amenable to an optimization approach, the problem of optimal platform design must first be cast in the standard form of an optimization problem, namely:

$$\underset{\mathbf{x}}{\text{minimize}} \quad f(\mathbf{x}) \qquad (2)$$

$$\text{Subject to} \quad g_i(\mathbf{x}) = 0 \; i = 1, n_e$$
$$g_j(\mathbf{x}) \le 0 \; j = n_e+1, n_e+n_i$$

where $\mathbf{x}$ is a vector of *design variables* which, in the present application, includes the major and minor semi-axes of the upper and lower platform (Ar_x, Ar_y, Br_x, Br_y), the gimbal attachment angles (α and β) and the 3 minimum leg lengths (Q_{min1}, Q_{min2}, Q_{min3}). The *objective function* $f(\mathbf{x})$ defines the scalar quantity which the designer is attempting to optimize. As will be detailed in the next section, the objective function was chosen to maximize the workspace in the various degrees-of-freedom. The *equality constraints* $g_i(\mathbf{x}) = 0$ normally represent interrelationships between the design variables which must be satisfied at the solution, but were not used in the present work (i.e. $n_e = 0$). Finally, the *inequality constraints* $g_j(\mathbf{x}) <= 0$ are used to place bounds on the design variables and functions of them. In the present work, these included bounds on the minimum dexterity of the motion-base, as well as on the minimum leg lengths. In a fully general *nonlinear programming problem*, the functions $f(\mathbf{x})$, $g_i(\mathbf{x})$ and $g_j(\mathbf{x})$ are completely general and have no particular form (e.g. linear, quadratic, convex, etc.). When this is the case, as in the present work, the optimization problem can be quite difficult to solve consistently and reliably.

There exist many techniques to solve general optimization problems.[10, 11] All of these techniques are iterative in nature - that is, they start from an initial guess for the solution, which is supplied by the user, and take steps toward to a local optimum which may or may not be the *global* optimum to the optimization problem. Thus, an important consideration in any minimization problem is the number of minima which the problem can be expected to have. In general, although there can only be one true global minimum, there may also be a number of local minima. This is illustrated in Figure 6 for the uni-dimensional case of an objective function f(x) which varies with a single design variable x. A result of this situation is that the solution algorithm may converge onto one of the local minima without detecting the global minimum. Our results indicated that our problem, as formulated, does indeed have multiple local minima. To circumvent this problem, a technique was implemented in which 70 different randomly-chosen initial starting points were used for each optimization problem solved. Each of these would converge onto a particular minimum and these were then sorted. Typically, 40 to 50 % of the starting points would converge onto the same best minimum found, and this gave some assurance that this should be the global minimum. However, it cannot be *proven* that this minimum was truly global due to the

complexity of the functions involved. The search for techniques which can consistently guarantee convergence to the global optimum for fully general optimization problems is still an active area of research.

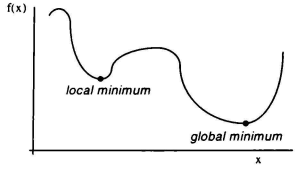


Figure 6 - The distinction between a local minimum and the desired global minimum

The size and direction of the steps taken toward the optimum are determined by the principle of operation of the particular technique used. Each solution iterate is checked against certain optimality conditions[12] which must be satisfied by the true solution to the problem. If these conditions are not satisfied within a user-supplied tolerance, another step is required. The computational cost of each iteration, as well as the convergence characteristics of the technique will determine the CPU time required to find the solution, as well as the variety of problems for which the algorithm will successfully converge. A comprehensive survey of evaluations of these techniques[11] concludes that overall, the sequential quadratic programming and the generalized reduced gradient techniques seem to work best for a variety of constrained nonlinear optimization problems. In the present work, the sequential quadratic programming method[12] was chosen. The principle of operation of the sequential programming technique is as follows: At each iteration, based on the current solution iterate, the algorithm formulates and solves a simplified problem whose solution is used as a step toward an improved solution iterate. This simplified problem consists of a quadratic objective function and linear constraints (i.e., a *Linearly-Constrained Quadratic Programming* sub-problem). The new solution is then checked against the optimality conditions for the true problem to determine whether another step must be taken.

Specification of the Objective Function

A hyper-ellipsoid, created by merging the two-dimensional mappings shown in Figure 4 into a six-dimensional mathematical function, describes the required motion volume. Its mathematical form is defined as,

$$\left(\frac{X-X_0}{\rho_x}\right)^2+\left(\frac{Y-Y_0}{\rho_y}\right)^2+\left(\frac{Z-Z_0}{\rho_z}\right)^2+\left(\frac{\psi-\psi_0}{\rho_\psi}\right)^2+\left(\frac{\theta-\theta_0}{\rho_\theta}\right)^2+\left(\frac{\varphi-\varphi_0}{\rho_\varphi}\right)^2 \le 1 \qquad (3)$$

where X_0,..., φ_0 denote the neutral position of the motion base (when all actuators are at their mid-stroke). This also corresponds to the midpoint of the hyper-ellipsoid. The *weighting factors* ρ_X,..., ρ_φ denote the ellipsoid semi-axis length in each direction. If the motion requirements are available as a set of trajectory maps (see Figure 4), the hyper-ellipsoid can be fitted to the data which contain the midpoints and weighting factors.

The objective function to be maximized is then specified as:

$$\overline{X}^2+\overline{Y}^2+\overline{Z}^2+\overline{\psi}^2+\overline{\theta}^2+\overline{\varphi}^2=R \qquad (4)$$

where $\overline{X}$, $\overline{Y}$,... are generalized coordinates, denoting a *scaled*, non-dimensional distance from the platform's neutral position, for example,

$$\overline{X}=\frac{X-X_0}{\rho_X} \qquad (5)$$

The factor R, which is called the *weighted radius of the hyper-ellipsoid*, scales the ellipsoid proportionally without changing its shape. The objective function given by eq.(4) attempts to find the *maximum weighted radius* R_{max}, describing the largest hyper-ellipsoid which just fits into the workspace of the motion platform. If $R_{max} \ge 1$, this implies that the workspace of the motion base will contain the design trajectories.

Because the motion-base must be symmetric about the X-Z plane, it follows that,

$$Y_0=\psi_0=\varphi_0=0 \qquad (6)$$

The X_0-Z_0 location of the platform reference indicates the neutral position with respect to the base, as well as the initial platform pitch angle θ_0. All of these values are determined during the optimization process.

Estimation of the maximum weighted radius R_{max}

Since there is no analytical method available to calculate R_{max}, a numerical approximation must be used. However the computational effort of this approximation should be kept within reasonable limits since each optimization loop typically requires a few hundred iterations to locate an optimum.

The approach used here is to restrict the analysis of the workspace to *cross-sectional planes* of the hyper-ellipsoid. For instance, in a three-dimensional case, an ellipsoid could be evaluated by restricting the analysis to its three cross-sectional planes, namely the X-Y, X-Z and Y-Z plane. In the six-dimensional case, there are 15 cross-sectional planes in the volume, including combinations of both translational and rotational directions: X-Y, X-Z, X-ψ, X-θ, X-φ, Y-Z, Y-ψ, Y-θ, Y-φ, Z-ψ, Z-θ, Z-φ, ψ-θ, ψ-φ and finally θ-φ.

This strategy reduces the estimation of the maximum weighted radius R_{max} to the estimation of 15 inscribed ellipses (i.e. determining the *local* maximum weighted radii R_{XY}, R_{XZ}, ... , $R_{\theta\varphi}$) in the cross-sectional planes of the workspace. The ellipse which yields the *smallest* value for R_{max} is considered critical, thus implying:

$$R_{max} \approx \min \{ R_{XY}, R_{XZ}, \ldots , R_{\theta\varphi} \} \qquad (7)$$

Figure 7 depicts this concept for three cross-sections of the *translational* workspace, namely X-Y, X-Z and Y-Z. Here, the weighting factors are $\rho_X=\rho_Y=1$ and $\rho_Z=0.5$, and the critical ellipse is located in the X-Z plane.

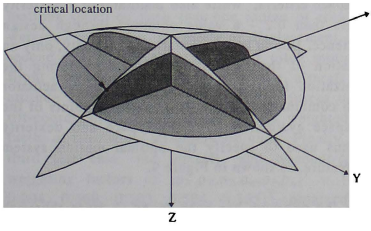


Figure 7: Three inscribed ellipses in cross-sectional planes of the workspace.

Determining the weighted radii of the inscribed ellipses

In order to define the inscribed ellipse for each cross-section, the boundary of the workspace in that cross-section must first be determined. Although an analytical determination of motion boundaries is possible,[13] it is restricted to translational motions, and is computationally intensive.

The numerical approach used here to find the boundary is, however, straight-forward, as there is a direct relation between the platform pose and the leg lengths. By moving the platform from its neutral position in a certain direction, the leg lengths are changed. At some point, one of these leg lengths will reach its specified length limit, thus restraining the platform motion.

The workspace boundary can be estimated by repeatedly searching for the boundary, each time in a different direction (the *offset* direction), while always starting from the same initial position. Figure 8 shows how a boundary estimate for the X-Z plane is found with 16 discrete offset directions.

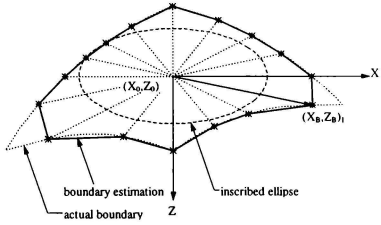


Figure 8: Estimate for X-Z motion boundary based on 16 discrete boundary searches.

The weighted radius of the maximum inscribed ellipse can be found by an appropriate choice of offset directions. This will now be described mathematically for the X-Z plane, using N_{bp} discrete boundary searches. First, equally spaced angles α_i are defined:

$$\alpha_i = \frac{i}{N_{bp}} \cdot 2\pi, \quad i = 1,\ldots,N_{bp} \qquad (8)$$

Next, an offset vector is defined for each direction:

$$\begin{bmatrix} \Delta X \\ \Delta Z \end{bmatrix}_i = \begin{bmatrix} \rho_X \cdot \cos(\alpha_i) \\ \rho_Z \cdot \sin(\alpha_i) \end{bmatrix} \qquad (9)$$

Then, the boundary points are detected by an iterative process, using the Jacobian matrix, as well as the actuator lengths and their limits. The process converges quadratically, normally yielding the boundary points in 2-3 iterations. One parameter k_i per direction defines the boundary points:

$$\begin{bmatrix} X_B \\ Z_B \end{bmatrix}_i = \begin{bmatrix} X_0 \\ Z_0 \end{bmatrix} + k_i \begin{bmatrix} \Delta X \\ \Delta Z \end{bmatrix}_i \qquad (10)$$

The accuracy of the workspace determination depends on the number of discrete boundary points which are analyzed per cross-section (N_{bp}) as well as the accuracy in the boundary point estimation itself (ε). In this work, we used N_{bp}=48 and $\varepsilon=10^{-5}$.

Finally, the weighted radius of the inscribed ellipse is defined as the minimum of all values k_i:

$$R_{XZ} = \min \{ k_1, k_2, \ldots, k_{Nbp} \} \qquad (11)$$

Applying eq.(7) then represents an approximation to the maximum weighted radius of the inscribed hyper-ellipsoid, R_{max}, because only the cross-sectional planes of the workspace are considered.

Dexterity

In the six degrees-of-freedom synergistic mechanism, six actuated links determine the position and rotation of the platform. By contracting and extending the links the final platform *pose* (position and rotation) is altered. This implies that there should be a one-to-one relation between the actuator link positions (the length of the actuators) and the platform pose. Problems can arise with the controllability of the platform when the platform pose varies substantially for very small motions of the actuators.

This situation is called a *singularity* of the platform and should be avoided throughout the workspace, as it can lead to malfunctions and failure of the platform. Furthermore, designs which yield nearness to singularities should be avoided, as they can result in excessively high actuator loads.

Singularities can be detected by analyzing the Jacobian matrix, which relates rates of change in platform pose to rates of change in actuator lengths:

$$\dot{\mathbf{q}} = \mathbf{J}\,\dot{\mathbf{x}} \qquad (12)$$

in which **q** denotes the actuator lengths and **x** denotes the platform pose, including both translations and rotations. The 6 x 6 matrix **J** is defined as:

$$\mathbf{J} = \begin{bmatrix} \frac{\partial q_1}{\partial x} & \cdots & \frac{\partial q_1}{\partial \varphi} \\ \vdots & & \vdots \\ \frac{\partial q_6}{\partial x} & \cdots & \frac{\partial q_6}{\partial \varphi} \end{bmatrix} \qquad (13)$$

Note that **J** is not constant, but varies throughout the workspace. The dexterity D is the reciprocal of the condition number of **J**, and can be defined as:

$$D = \frac{\sigma_{min}}{\sigma_{max}} \qquad (14)$$

in which σ_{min} and σ_{max} denote the minimum and maximum singular value of **J**, obtained by a process called Singular Value Decomposition.

The dexterity indicates the controllability of the platform. A value of D=1 indicates an isotropic condition, in which *equal* joint effort is needed to obtain motion in *every* direction. When approaching a singularity of the platform, σ_{min} will tend to zero, and thus a value of D=0 will indicate a singular configuration.

Optimizing the Motion-Base Architecture

The optimization process attempts to define a motion system architecture which achieves the desired workspace criteria, while also maintaining a dexterity no less than 0.2 - a value obtained through design experience. Due to the computational intensity, only an estimation of the dexterity is calculated during the optimization (at the 64 minimum/maximum actuator length combinations and at the boundary points of the workspace evaluation). The workspace and dexterity are thus used to specify the optimal motion system architecture, as shown in Figure 9.

Once the optimum has been found, a more thorough check of the minimum dexterity is performed, by dividing each of the actuator lengths into ten sections, and evaluating the dexterity at each of the 10^6 combinations. During these calculations, the minimum absolute distance between all the legs is also computed. If this minimum ever falls below a specified level (in this case 15 cm), the program indicates that

there is a potential *leg-crossing* situation and the candidate geometry is rejected.

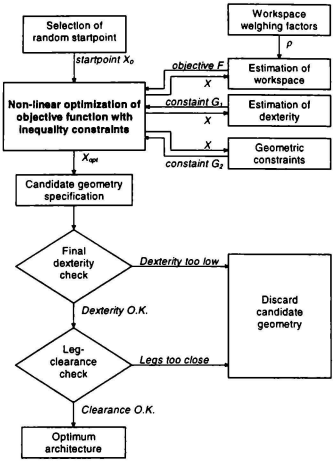


Figure 9 - Simulator motion-base architecture optimization procedure

Test Configurations

Three test cases are now shown to illustrate the use of the design software. In the first test case, the weighting factors in all degrees-of-freedom are equal, and the relationship between the angular and rectilinear motions specified such that one radian of rotation is weighted equally to one meter of displacement. In the second test case, since the use of pitch motions in flight simulators re-creates both the aircraft pitch attitude as well long-duration specific forces through tilt-coordination, the pitch motion is weighted by a factor of two, while all others are unity. Finally, in the third test case, the aircraft- and washout-specific weighting factors of $[\rho_X,\rho_Y,\rho_Z,\rho_\psi,\rho_\theta,\rho_\phi]^T = [0.821, 0.204, 0.369, 0.103, 0.504, 0.355]^T$, obtained from Figure 4, are used.

Discussion of Results

Table 1 gives the final values of the design variables and R_{max} at the global optimum for each of the three test cases. The salient points in this table are

- the upper platform is circular or near-circular in all test cases, and the semi-axes lie against their minimum constraint of 1.5 m.
- the lower platform is always elliptical, though its major semi-axis lies in the x-direction in the first two test cases, but in the y-direction in the third test case.
- the angle β lies against its constraint of $\pi/2$ rad in the second test case.
- R_{max} is less than unity in the first two test cases, indicating that only a scaled-down hyper-ellipsoid was achievable.
- R_{max} is greater than unity in the third test case, indicating that the Boeing 747-400 trajectories are achievable with this motion base geometry.

	Unity Weighting	Enhanced Pitch	Boeing 747-400
Ar_X [m]	1.500	1.500	1.500
Ar_Y [m]	1.500	1.581	1.500
Br_X [m]	2.544	2.415	1.837
Br_Y [m]	1.808	2.055	2.130
α [rad]	2.156	1.963	2.155
β [rad]	1.770	1.571	1.853
Q_{min1} [m]	2.203	2.437	2.213
Q_{min2} [m]	2.139	1.987	2.042
Q_{min3} [m]	2.051	1.970	2.138
R_{max}	0.414	0.304	1.014

Table 1- Values of the design variables and objective function for the optimum geometries.

Figure 10 shows the resulting mechanisms graphically as well as their translational and rotational workspaces. The first column, showing the results of the uniformly-weighted ellipsoid, has the most uniform workspace of the three geometries. However the layout of the mechanism deviates considerably from the standard Stewart Platform. When the pitch weighting is doubled, the mechanism in the second column is obtained. The rotational workspace shows a distinct elongation along the pitch axis, achieved with a rather unconventional form of the mechanism. Finally, using the weighting function derived from the predicted

trajectory of the 747-400 aircraft, the mechanism in column 3 is obtained. Its workspace shows that the x-direction motions are the most emphasized, relative to all others. While this mechanism does not deviate drastically from conventional designs, it offers optimum cueing capabilities, based on the washout filter used. Moreover, the forward gimbals are lower than the aft pairs, which would allow easier placement of the visual display system with respect to the pilot eye position. In fact, this architecture lends itself well to a full-flight simulator.

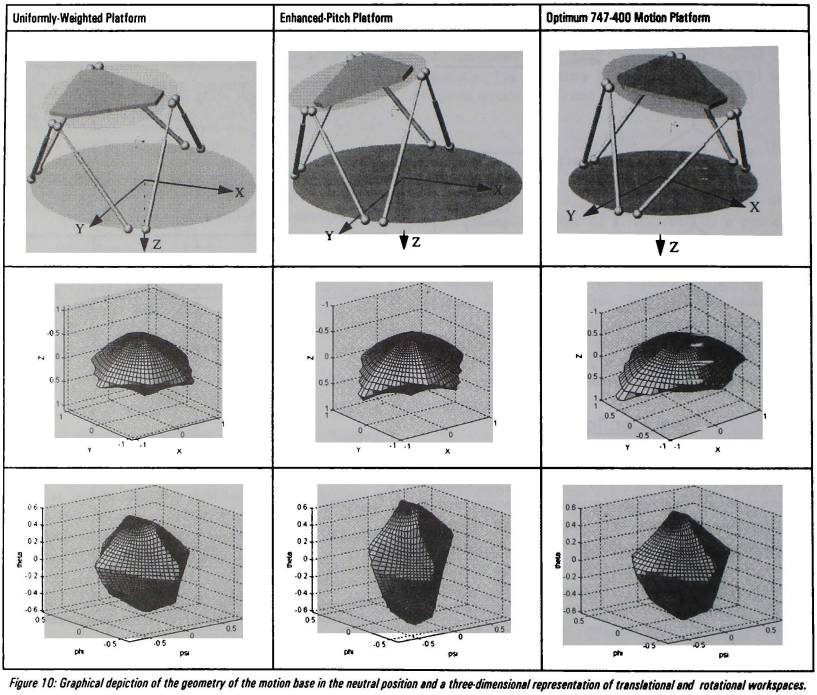


Figure 10: Graphical depiction of the geometry of the motion base in the neutral position and a three-dimensional representation of translational and rotational workspaces.

Figure 11 shows a subset of the 15 planar cross-sections through the six-dimensional workspace of the three platforms, as well as the corresponding maximum inscribed ellipses. The doubled weighting on pitch is clearly evident in the sections of column 2, while the non-uniform weighting of the third test case is evident in column 3.

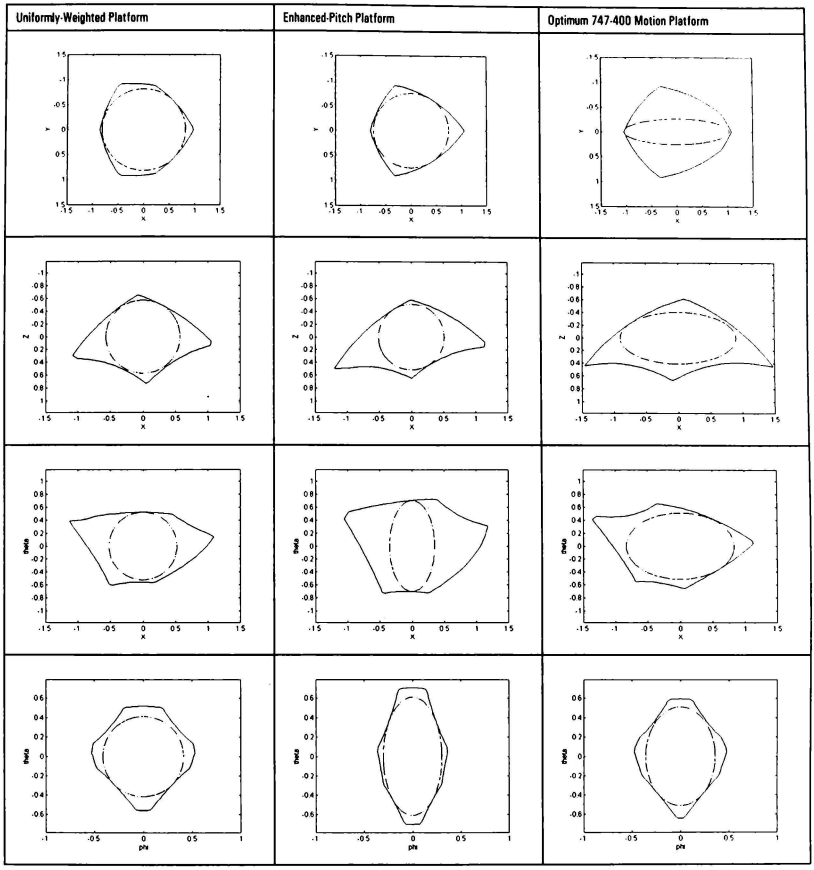


Figure 11: Cross-sections of the workspace in X-Y direction, X-Z direction, X-θ direction and φ-θ direction and inscribed ellipses for these cross-sections

The goal set forth in the beginning of this research was to tailor the workspace available by manipulating the design variables, while remaining within the specified constraints. The freedom to allow the upper and lower attachment points to fall on elliptical, rather than on circular lines, was found to yield a significant advantage. This does, however, require unequal actuator lengths. The costs related to this non-uniformity can be reduced by developing actuators with similar hydro-mechanical component geometries, while allowing only the piston-cylinder lengths to vary.

Furthermore, this approach allows the realization of mechanisms as functions of the cueing requirements, rather than trying to work around the workspace constraints of standard Stewart Platforms through

excessive attenuation of the aircraft motions, or the use of non-linear adaptive filters.

Conclusions

The design of a flight simulator is a highly-integrated multi-disciplinary activity with the eventual goal of providing realistic cues to the simulator pilot. This paper re-considers the conventional motion system design procedure, by addressing the cueing requirements *prior* to specifying the motion system, and then defining an appropriate motion system geometry. The flexibility offered by this technique allows tailoring the motion system to particular needs. The optimization approach adopted here emphasizes that the specification of a motion platform geometry need not be restricted to the conventional Stewart Platform.

While this technique thoroughly examines the *kinematic* aspects of motion bases, further analysis should follow. The remaining task involves designing the crew module, visual display system, and the structure to integrate all on-board systems with the moving platform. Then, with knowledge of the mass and stiffness properties of the entire moving load, detailed dynamic analyses can be performed.

It should be highlighted, however, that the platforms of multi-purpose facilities, such as the SIMONA Research Simulator, often require a wide range of workspace capabilities, as well as high bandwidth. In these instances, a generic Stewart Platform may be the optimal choice, given the wider range of applications required than with a training simulator.

Acknowledgments

This paper overviews part of the first author's doctoral research. The Flight Training Centre of KLM Royal Dutch Airlines is thanked for their input in this research, particularly with regard to the use of their facilities for the generation of the motion envelope of the 747-400 aircraft. The sabbatical stay of Dr. Nahon was partly funded by the Netherlands Scientific Agency NWO. Mr. Herold Cremer is thanked for the creation of the motion-base CAD drawings.

References

[1] Stewart, D., "A Platform with six-degrees-of-freedom", in Proc. Inst. of Mechanical Engineers, vol. 180, part 1, no. 5, 1965-1966, pp. 371-386.

[2] Advani, S.K. et al, "SIMONA - A Reconfigurable and Versatile Research Facility". AIAA Modeling and Simulation Technologies Conference, New Orleans, August 1997, AIAA-97-3809-CP.

[3] Nahon, M.A., Reid, L.D. "Simulator Motion-Drive Algorithms: A Designer's Perspective". Journal of guidance, AIAA, Vol.13, No. 2, 1989.

[4] Ish-Shalom, J., "Design of Optimal Motion for Flight Simulators". Ph.D. dissertation, Massachusetts Institute of Technology, 1982.

[5] Martin, E.A., "Force and Motion Cueing". In Flight Simulation Update, American Inst. of Aeronautics and Astronautics, January 1996.

[6] Grant, P.R., Reid, L.D., "Motion Washout Filter Tuning: Rules and Requirements" In AIAA Flight Simulation Technologies Conference, AIAA-95-3408-CP, Baltimore, MD, August 1995.

[7] Nahon, M.A., Ricard, R., and Gosselin, C.M., "A Comparison of Flight Simulator Motion-Base Architectures". From *Making it REAL*, CEAS Symposium on Simulation Technologies, Delft , The Netherlands, October 1995.

[8] Nahon, M. and Reid, L.D., "Simulator Motion-Drive Algorithms: A Designer's Perspective". Journal of Guidance, AIAA, Vol. 13, No. 2, 1989, pp. 356-362.

[9] Reid, L.D., and Nahon, M., "Response of Airline Pilots to Variations in Flight Simulator Motion Algorithms". AIAA Journal of Aircraft, Vol. 25, No. 7, July 1988, pp. 639-646.

[10] Gill, P., Murray, W., and Wright, M.H., Practical Optimization, Academic Press, 1981.

[11] Reklaitis, G.V., Ravindran, A., and Ragsdell, K.M., Engineering Optimization: Methods and Applications, John Wiley and Sons, New York, 1985.

[12] Schittkowski, K., 'NLPQL: a Fortran Subroutine Solving Constrained Nonlinear Programming Problems', Annals of Operations Research, Vol. 5, pp. 485-500, 1985.

[13] Gosselin, C., "Determination of the Workspace of 6-DOF Parallel Manipulators", ASME Journal of Mechanical Design, Vol. 112, No. 3, 1990, pp. 331-336.

IS THERE AN OPTIMUM MOTION CUEING ALGORITHM ?

Weimin Wu
Graduate Student

Frank M. Cardullo
Associate Professor of Mechanical Engineering

Department of Mechanical Engineering
State University of New York
Binghamton, N. Y.

Abstract

Research was conducted by the authors to develop the optimum Cueing algorithm for a synergistic six degrees of freedom platform motion system. Three types of Cueing algorithms, classical, adaptive and optimal, were investigated. The classical algorithm performed more poorly than the others. Some improvements were made to the adaptive algorithms such as reducing the magnitude of false cues. The optimal algorithm was found to have the potential to be improved and was redesigned subsequently. A design approach featuring Fortran/Matlab/Simulink interactive design was used, which made it possible to try hundreds of sets of parameters. Comparisons were made mainly between the improved adaptive algorithm and the new optimal algorithm. The evaluation was based on the implementation of a model of the human's vestibular system to calculate the perceptual error between the aircraft and the simulator. Results showed that the optimal algorithm has some advantages over the adaptive algorithm and might be the best among the algorithms involved in this study. At the same time the adaptive algorithm was shown to be also a well developed algorithm.

I. Introduction

With the continuing improvement in hardware and software, flight simulation is playing an expanding role in the training of aircraft crews, the design of new aircraft, and entertainment. This study was conducted to provide a drive algorithm for a modern synergistic six degrees of freedom motion system for an aircraft simulator. Three types of algorithms, classical, adaptive and optimal algorithm were considered in this study. These motion base drive algorithms were first investigated and then improved and to some extent redesigned.

Copyright © 1997 by the American Institute of Aeronautics and Astronautics, Inc. All rights reserved.

The goal of the motion system is, along with the visual system, to provide motion sensations to the pilot so that in a simulator the pilot performs the controls and maneuvers in a manner which is consistent with how they would be performed in a real aircraft. When motion was provided, there was an increase in the occurrence of high-frequency/low amplitude control movements[17]. These changes served to make the control activity in the moving simulator appear more like that observed during flight than that recorded in static simulator. The presence of motion was found to reduce phase lag, increase the mid-frequency gain and cross-over frequency, and reduce the size of the remnant. When the simulated vehicle became more unstable, maneuver motion became important to control, its presence allowed the operator to exercise control even in regions where control by visual cues alone would be impossible.

Since a ground-based flight simulator system cannot duplicate the motions of an actual aircraft, it becomes necessary to develop a motion drive algorithm which utilizes the simulator's limited capabilities to provide the most necessary and beneficial motion cues. The motion base driving algorithm is also called a Cueing algorithm. It is also critical for the Cueing algorithm to avoid any improper motion cues since it is commonly known that improper motion cues in some flight conditions have negative effects on the simulation. It's reported that some motion systems of aircraft simulators have been turned off to avoid improper motion cues. Since modern motion system hardware seems to be capable of the task, the problem of inappropriate motion cues is considered to be largely a problem with the Cueing algorithm. The implemented Cueing algorithm will be used to drive the McFadden 676B-B046 simulator at the NASA Langley Research Center.

The typical, affordable platform device will have relatively limited displacement capability, and will

therefore be limited regarding the duration of its displayed cues. Washout is a commonly used aircraft simulation technique which separates the aircraft motion cues into high and low frequency components and simulate them by different means so that cues can be displayed within the physical confines of a given platform system. Washout may also scale down the motion which is displayed. Washout entails 'sneaking' the cab back towards a neutral or steady-state position following the display of the "onset" portion of a motion cue; i.e., the displacement and velocity resulting from the display of a cue are 'washed out' (hopefully at levels below that of pilot awareness). Washout provides some form of high pass filtering to limit the simulator cab excursions. The form of which may be linear or non-linear. Non-linear designs have included adaptive filters and other optimal control techniques applied against various criteria.

The only way in which a sustained specific force cue can be presented using the conventional motion platform is to tilt the platform, and allow the gravity vector to provide the sustained cue. This technique is known as gravity alignment. Gravity-alignment techniques exploit the inability of the otoliths to distinguish between pitch (or roll) and longitudinal (or lateral) specific force. The trick is to present these sustained cues while maintaining any false platform angular rate levels (i.e., angular rates not associated with angular onset cues) below the threshold of the semicircular canals. The surge and pitch degree-of-freedom are usually grouped as the surge/pitch channel and the sway and roll degree-of-freedom are grouped as the sway/roll channel so that gravity alignment can be employed to provide sustained specific force cue while motion cue relationships among the various degrees-of-freedom can be preserved. Conventional motion platform can not provide sustained heave-axis motion cues.

II. Description of the Cueing algorithms

Four washout algorithms are included in this study. The first is the so called classical Algorithm which is a derivative of the design of Schmidt and Conrad in the early 1970s[19] . The second algorithm evaluated in this study is the "Coordinated adaptive Washout Algorithm" developed by Parrish, Dieudonne and Martin at the NASA Langley Research Center in the early to mid 1970s[4]. The next cueing algorithm reviewed in the study is a variation of the Coordinated adaptive Washout Algorithm. This was developed by Prof. L. Reid and M. Nahon at the University of Toronto in the 1980s[3]. The optimal algorithm, the fourth algorithm employed in this study is that developed at MIT by Sivan et al[5] and implemented in the 1980s as described in ref. 2 & 3. All the Cueing algorithms has a similar architecture which is composed of separate filters for the translational degrees of freedom and the rotational degrees of freedom with a crossover path to provide the steady state or gravity alignment cues. Figure 1 shows this architecture.

In fig.1, $\underline{a}_{desired}$ and $\dot{\underline{\beta}}_{desired}$ are the translational acceleration and the angular rate which if applied to the simulator, the motion sensation in the aircraft will be replicated in the simulator. All the four algorithms have fixed filters in on-line simulation. The classical algorithm has fixed gains and manually constructed filters. The gains and filters are by no means optimized. The adaptive algorithms have adaptive gains and manually constructed filters. The algorithm attemps to optimize the gains continuously by minimizing the instantanious value of a cost function which restrains both output error and the magnitude of simulator states. The optimal algorithm has fixed gains (linear or nonlinear) and optimal filters which are optimized by an off-line optimal algorithm. The criterion for the optimization is to minimize the value of a cost function which restrains both sensation error and the magnitude of simulator states. A model of the human's vestibular system was involved in the off-line optimal algorithm design.

Theoretically the adaptive algorithms and the optimal algorithm have some advantages over the classical algorithm. The adaptive algorithms continuously adjust the adaptive gains according to the current states of the simulator and the aircraft input. In this way it should be possible to make full use of the simulator's motion system at all time. The optimal algorithm theoretically optimizes the control filters which should have better performance than the intuitively constructed filters.

III. Results

A. Initial results

A variety of inputs were employed to drive the four washout algorithms. The outputs showed that all the algorithms achieved the basic functionality of a Cueing algorithm. All the algorithms attributed high-pass properties to the translational channel and entailed 'sneaking' the simulator cab back towards a neutral or steady-state position following the display of the 'onset' portion of a motion cue. All the algorithms implemented

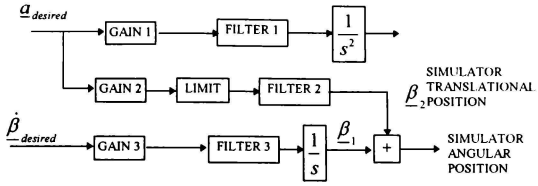


Figure 1: Illustration of the General Architecture of a Cueing Algorithm

cross-over channels which attempt to achieve gravity alignment to provide a sustained translational motion cue. The classical algorithm performed more poorly than the other three algorithms. The adaptive algorithms had some obvious problems most of which were fixed in this study. The UTIAS adaptive algorithm strove for more flexibility, but results showed that it did not behave better than the NASA one while it had some more undesirable properties. Major attention was paid to the NASA adaptive algorithm rather than to the UTIAS adaptive algorithm in the next phase of study. The optimal algorithm had some very attractive properties but resulted in excessive actuator extensions, which were attributed to the different selection of center of rotation.

B. Improvement of adaptive algorithms

Some significant spikes occurred when the adaptive algorithms were run (fig.2). Whenever there was a sharp change on the translational acceleration input, a spike would occur on the specific force output. After

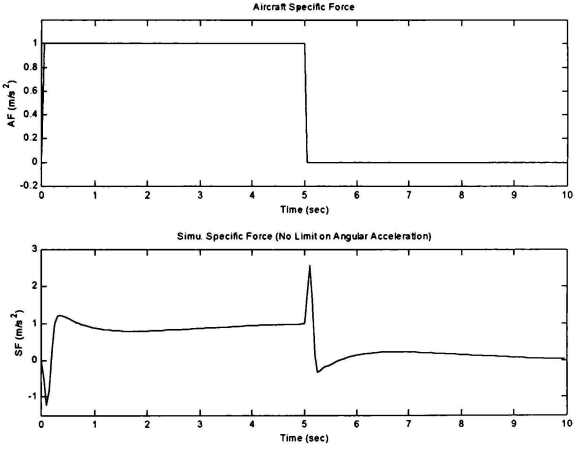


Figure 2: Specific forces at aircraft pilot's head and simulator pilot's head generated by the NASA adaptive algorithm with no limit on angular acceleration. Input: longitudinal acceleration, pulse input 1 m/sec^2 x 5 sec.

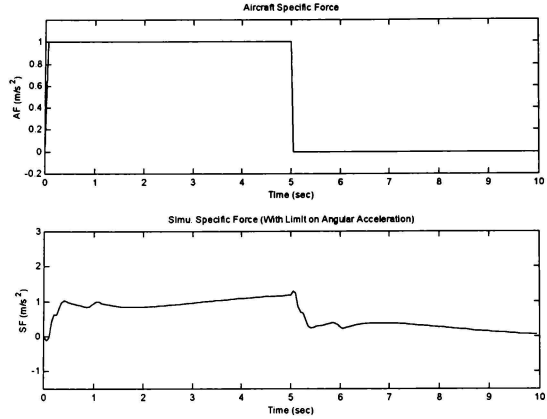


Figure 3: Specific forces at aircraft pilot's head and simulator pilot's head generated by the NASA adaptive algorithm with a limit on angular acceleration. Input: longitudinal acceleration, pulse input 1 m/sec2 x 5 sec. lim($\ddot{\theta}$) = 0.5 rad/sec^2.

careful examination, the cause of the spikes was identified to be the rotational acceleration of the simulator. An intuitive approach, to add a limit on the tilting angular acceleration, was employed to decrease the magnitude of the spikes. Spikes were attenuated significantly at the price of suffering a little more delay of simulator tilt response (fig.3).

In some rotational input cases, the adaptive algorithms generated some extra oscillations. The adaptive algorithms tried to eliminate the difference between the gravity vector in the aircraft and the gravity vector in the simulator. It was found in this study that the difference could not be eliminated since usually the simulator rotates an angle smaller than the aircraft does and the simulator translational acceleration could not be sustained for a long time. The simulator was trying to do something beyond its capability and oscillated around its neutral position. Detailed analysis can be found in ref.1. Some revision was made to the adaptive algorithms. The revised algorithm generates the same results as those generated by the old adaptive algorithms under translational inputs. The difference is that the revised version has a null translational channel under rotational input. The revised algorithm does not have the undesirable oscillations under translational, rotational, or a mixture of translational and rotational inputs .

C. Redesign of the optimal algorithm

The optimal algorithm was re-designed and the center of simulator rotation was re-defined. More terms were added to the optimal algorithm cost function to yield more flexibility on tuning the algorithm. A nonlinear gain which adjusts itself according to the current aircraft input was developed to give the optimal algorithm some nonlinear properties. The optimal algorithm selected the pilot's head as the center of simulator rotation to avoid any cross-coupling from simulator rotation to pilot's head translational motion. It was found in this study that this selection of center of rotation caused the excessive actuator extensions. Detailed analysis can be found in ref.1. The adaptive algorithms and the classical algorithm selected the centroid of the simulator as the center of rotation. Cross couplings were ignored in the algorithm development. Test results showed that some undesirable spikes were generated due to the cross-couplings. Some effort was spent on reducing the magnitudes of the spikes as discussed before. The spikes were reduced but not eliminated finally. The

spikes represent some false motion cues. A new optimal algorithm which chose the centroid of the simulator as the center of rotation and explicitly expressed the cross-couplings was then developed in an attempt to achieve a better washout algorithm.

More terms were added to the optimal algorithm cost function to yield more flexibility on tuning the algorithm. All the system matrices were constructed in Matlab functions. The algebraic Riccati equation was also defined in Matlab functions. The definition was transported to a Fortran environment in which the algebraic Riccati equation would be solved by some Fortran routines. The solution of the algebraic Racati equation was transported back to the Matlab environment and some Matlab functions would construct the desired optimal filters. The symbolic calculation involved in the old optimal filter design procedure was bypassed (details in ref.1). A Matlab function would cancel common poles and zeros and yield optimal filters in reduced order. A simulation set-up which read in those transfer functions automatically was constructed in Simulink . The effects of the optimal filters could be visualised by simulation in Simulink. If the filters are not satisfactory, the procedure of design would be repeated by selecting some new cost function parameters, until a satisfactory result could be finally approached. After selection of a set of parameters, one cycle of the whole design procedure cost only 30 seconds while old design approaches would require more than 15 minutes. The new design approach made it possible to try hundreds of sets of parameters. As a result, the optimal algorithm was well tuned.

D. Final results

When the pitch/surge or roll/sway channel was tested, the specific force output of the NASA adaptive algorithm often went in the wrong direction at the beginning of a translational acceleration input. Although some improvement was made to reduce the magnitude of the spikes, it was found that the magnitude and duration of the specific force in the wrong direction were still too large to be ignored when the input contained some frequency components near or higher than 0.5 Hz . It is found that the optimal algorithm successfully handled this problem. Results showed that the specific force in the wrong direction was completely eliminated. The optimal algorithm also generated specific force outputs with significantly less shape distortions than the NASA adaptive algorithm did in the pitch/surge or roll/sway test cases. In the heave test case, the NASA adaptive algorithm didn't generate an obvious specific force drop when there was a large drop of the input while the optimal algorithm did, which was obviously desired. When the pitch or the roll channel was tested, the NASA adaptive algorithm generated simulator angular velocity output with very nice shape while the output of the optimal algorithm had some visible distortion. The optimal filter for the pitch channel was designed using the pitch resulting from the gravity alignment due to a surge input. It is expected if the pitch only input was used as well, then a better filter design would result. This is also true for the filter for the roll channel. When the yaw channel was tested, the output of the NASA adaptive algorithm and the optimal algorithm were different. If the non-linear gains for the two algorithms are adjusted so that the two algorithms drive the simulator to the same amount of actuator extensions, the simulator angular velocity output of the NASA adaptive algorithm will have higher magnitude but be sustained for shorter time than the output of the optimal algorithm. It's not clear, at this point, which output will result in better simulation effects.

It was found that in most input cases, to generate the specific force at the simulator pilot's head or the simulator angular velocity with the same magnitude, the NASA adaptive algorithm and the optimal algorithm would result in about the same amount of actuator extensions.

Some of the outputs are shown in fig. 4.

IV. Conclusion and Recommendations

The classical algorithm performed more poorly and had no theoretical advantages over the other algorithms. The adaptive algorithm and the optimal algorithm performed well generally. The optimal algorithm performed better than the adaptive algorithm. The advantages of the optimal algorithm over the adaptive algorithm are that the optimal algorithm eliminated the false motion cues, generated outputs with better shapes in many simulation cases, and did not lose some desirable motion cues while the adaptive algorithm did in some input cases. The filters of the optimal algorithm are optimized theoretically while those of the adaptive algorithm are not. The adaptive gain is still an advantage of the the adaptive algorithm did in some input cases. The filters of the optimal algorithm are optimized theoretically while those of the adaptive algorithm are not. The adaptive gain is still an advantage of the adaptive algorithm

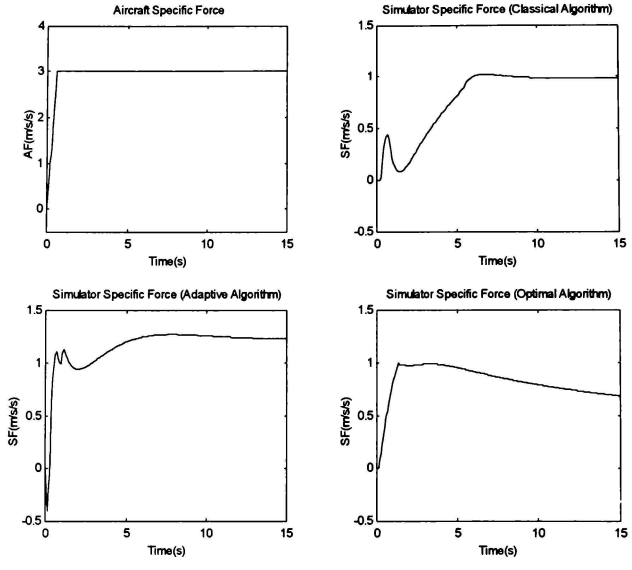


Figure 4: Comparison of the Specific Force Generated by Different Algorithms. Input: Ramp to Step Input.

over the optimal algorithm although the nonlinear gain implemented for the optimal algorithm helped cut the drawback of its linear property. Currently the optimal algorithm appears to be the best of the algorithms involved in this study. There might be some ways to design a better algorithm. One way recommended by the authors is to replace the filters of the adaptive algorithm with the optimized filters generated by the off-line optimal algorithm. Then re-construct the control laws for the adaptive algorithm. The other way is to design some on-line optimal algorithm which will optimize the control filters continuously according to the aircraft input and the simulator states.

References

1. Wu, W., Development Of Cueing Algorithm For The Control Of Simulator Motion Systems. Thesis for MS. SUNY at Binghamton, May 1997.
2. Reid, L. D. and Nahon, M. A., "Flight Simulation Motion-base Drive Algorithms: Part 1 - Developing and Testing the Equations." UTIAS Report No. 296, CN ISSN 0082-5255, December 1985
3. Reid, L. D. and Nahon, M. A., "Flight Simulation Motion-base Drive Algorithms: Part 2 - Selecting the System Parameters." UTIAS Report No. 307, CN ISSN 0082-5255 , May 1986
4. Martin, D. J., Jr. , "A Digital Program for Motion Washout on Langley's Six-degree-of-freedom Motion Simulator." NASA Contractor Report 145219 Contract NAS1-14600, July 1977
5. Sivan, R. , Ish-shalom, J. and Huang J. K. "An Optimal Control Approach to the Design of Moving Flight Simulators." IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-12, No.6, November/December 1982. P 818-827.

6. Conrad, B. , and Schmidt, S. F. ; Douvillier, J. G., "Washout Circuit Design for Multi-degrees-of-freedom Moving Base Simulators." AIAA Paper No. 73-929, AIAA Visual and Motion Simulation Conference, September 1973.
7. Ogata, Katsuhiko, *Modern Control Engineering* (Second Edition) , N.J.: Prentice Hall, 1990.
8. Goldberg, J. M. and Fernandez C. : Physiology of Peripheral Neurons Innervating Semicircular Canals of the Squirrel Monkey. Journal of Neurophysiology. Vol 34. No.4, 1971.
9. Goldberg, J. M. and Fernandez , C. , Journal of Neurophysiology, Vol. 39. No.5 , 1976. P 971-1009 .
10. Ghista, D. N. , Schmid, R., Buizza, A. and Zambarbieri, D., Applied Physiological Mechanics Vol.1, OPA, Amsterdam, B.V., 1979. P 779-893.
11. R.J.A.W. Hosman and J.C. van der Vaart : Vestibular Models and Thresholds of Motion Perception, Results of Tests in a Flight Simulator. Delft University of Technology, Department of Aerospace Engineering . Delft, the Netherlands, April 1978.
12. Zacharias, G. L. : Motion cue models for pilot-vehicle analysis. Defense Documentation Center. Defense Logistics Agency. Cameron Station. Alexandria, Virginia. May, 1978.
13. Young, L. R. and Oman, C. M. : Model for Vestibular Adaptation to Horizontal rotation. MIT, Cambridge, Massachusetts. 1969.
14. Clark, B. and Stewart, J. D. : Effects of Angular Acceleration on Man: Thresholds for the Perception of Rotation and the Oculogyral Illusion. Aerospace Medicine. September, 1969.
15. Clark, B., Stewart, J. D. and Nathan, H. P.: Thresholds For Detection Of Constant Rotary Acceleration During Vibratory Rotary Acceleration. Aviation, Space, and Environmental Medicine. June, 1980.
16. Young, L. R. : Perception of the Body in Space: Mechanisms. MIT, Cambridge, Massachusetts. 1982.
17. Gundry, A. J. : Thresholds to Roll Motion in a flight Simulator. AIAA Visual and Motion Simulation Conference , dayton, Ohio, April, 1976.
18. R.J.A.W. Hosman and J.C. van der Vaart : Thresholds Of Motion Perception And Parameters Of Vestibular Models Obtained From Tests In A Motion Simulator. Effects Of Vestibular And Visual Motion Perception On Task Performance. Delft University of Technology, Department of Aerospace Engineering . Delft, the Netherlands, April 1978.
19. Martin E. A. : Motion and Force Simulation Systems I, Whole-body Motion Simulators. 13th Annual Flight & Ground Simulation Update, State University of New York at Binghamton. January 1997.

Comanche Player Station

Comanche Simulation in the Aviation Warfighting Cell

Chris Romanoff
Senior Systems Engineer
Sikorsky Aircraft Simulation Group

1.0 ABSTRACT

Modeling and Simulation (M&S) is used extensively in the acquisition process of the RAH-66 Comanche aircraft program. Specific applications include the augmentation of: system testing, engineering development, war-fighting development, and training. It is critical that a simulator system accurately represent the designed Comanche capabilities when participating in advanced war-fighting experiments (AWE) and advanced technology demonstrations (ATD).

The Comanche Player Station (CPS) was built by Sikorsky Aircraft under the RAH-66 Comanche contract. This device is a fully functional simulator that was developed for the United States Army. CPS is a key component of the Aviation War-fighting Cell (AWC) which is part of the Army's Battle Lab network. It recently participated in an AWE known as the Anti-Armor Advanced Technology Demonstration (A²ATD), Experiment #5. This experiment achieved the successful integration of Comanche and Longbow Apache man-in-the-loop simulations performing cooperative missions on a digital battlefield environment.

Extensive use of the CPS will be made by the Army in future years. Early Operational Capability (EOC) pilots plan to use the device to develop tactics, techniques, and procedures (TTP) and also perform Comanche system familiarization. Numerous experiments are planned involving Comanche on the digital battlefield. CPS is a compact high fidelity simulation that the Army is presently using. It's purpose is to assist in the development of a production weapon system that will not be available until 2006.

CPS hardware and modular software architecture represent the latest in simulation technology. The device is Distributed Interactive Simulation (DIS) compliant and is presently operating in the AWC at the Aviation Test Bed (AvTB) at Fort Rucker, Alabama.

2.0 INTRODUCTION

The RAH-66 Comanche is the new reconnaissance and light attack helicopter that is being developed for the United States Army. This aircraft will represent the latest in rotorcraft technology. Comanche is being designed with major improvements in several key areas, these include: a light weight composite airframe structure, embedded Fantail™ anti-torque system, bearing-less main rotor system, reduced infra-red (IR) signature, and fly-by-wire electronic flight control system. The target acquisition system will be the most advanced system available on any Army helicopter. Comanche crew systems include a high performance helmet integrated display and sighting subsystem (HIDSS) that will provide flight critical symbology to the pilot, as well as FLIR imagery at night. The all-glass cockpit consists of multi-functional displays that provide several layers of information to the pilot. The Comanche cockpit's integrated functionality is significantly different than that of existing aircraft with conventional instruments.

Simulation has played a major role in the acquisition and development strategy of the Comanche aircraft. The program has used simulation from the initial stages of concept formulation to the current early operational capability (EOC) design.

In 1983, simulation was used in a series of feasibility studies, alternative concept evaluations, and trade-off analyses to determine the need for a new aircraft system. Product specification and operational system requirements were also defined during this period. In 1991, following contract award to Boeing-Sikorsky, simulation was used extensively during the Comanche demonstration/validation (DEM/VAL) phase. During

Copyright © 1997 by the American Institute of Aeronautics and Astronautics, Inc. All rights reserved.

this period, a significant effort was made to design and evaluate several areas of the aircraft. These included: the crewstation interface, flight controls and air-vehicle handling qualities, mission equipment design; and several other risk reduction efforts.

From the Comanche's initial conception, the Army has focused on the mission effectiveness of the aircraft as a future weapons platform, and simulation has played a major role in the concept to production process thus far. Simulation is a critical element in the process leading to the overall design, development, and operational effectiveness of this advanced weapon system.

In 1993, the focus began to shift towards evaluating the mission effectiveness of the Comanche. The basic issue was how the Comanche would best be utilized in the battlefield environment. The need to develop a set of tactics, techniques and procedures became apparent. Without an actual aircraft to use, the Army turned to simulation to help get some of the answers. The Comanche Engineering Development Simulator (EDS) at Sikorsky Aircraft in Stratford, Connecticut has been used as a full mission simulator since 1986. The task to be accomplished in 1993 was to get the Comanche weapon system simulation into an operational environment. That environment, known as the "Digital Battlefield", consists of several simulated and live entities that are linked together by a wide area network (WAN). The WAN used for these exercises is known as the distributed simulation internet (DSI). The DSI is a high speed network that provides a link to simulation facilities through a strategically distributed hub network. Players in the digital battlefield consist of manned simulators, such as the Comanche EDS, computer generated forces (CGF), and live weapons platforms in the field. The DSI network provides connectivity to each device. The communications protocol that allows for inter-connectivity is the distributed interactive simulation (DIS) protocol.

In May 1993, the Comanche EDS introduced to the digital battlefield at the Association of the United States Army (AUSA) symposium in Orlando, Florida. Located in Connecticut, the Comanche EDS was digitally linked with several other simulators across the country. These included: a Comanche-like simulator at the Crew Station Research and Development Facility (CSRDF) at NASA Ames Research Facility in Moffett Field, California, Manned tank simulators at the Tank and Automotive Research and Development Engineering Command (TARDEC) in Warren, Michigan, and manned AH-64 Apache simulators at the Aviation Test Bed located at Fort Rucker, Alabama. All of these "entities" were digitally networked and displayed on the AUSA show floor in Orlando, Florida. The computer generated forces, both friendly (blue) and threat (red) were generated by a tactical scenario simulation provided by CAE-Electronics and known as ITEMS™. This exercise was extremely successful, and provided the Comanche Program and Boeing-Sikorsky valuable insight to the power of distributed interactive simulation. From that initial AUSA demonstration, Boeing-Sikorsky and the Comanche Program have brought this type of simulation to AUSA Symposiums in Washington, DC in October 1993 and 1994.

3.0 AVIATION WARFIGHTING CELL (AWC)

There are three primary battle-lab facilities in the Army that are the focus of all DIS activity. They are located at Fort Benning, Georgia, Fort Knox, Kentucky, and Fort Rucker, Alabama. The battle lab at Fort Rucker is known as the Aviation Test Bed (AvTB). Within this aviation battle lab, the Army has developed the Aviation War-fighting Cell (AWC). The primary purpose of the AWC was to bring advanced aviation components to the "digital battlefield". AWC was developed to support upcoming advanced war-fighting experiments that require an aviation component.

The first experiment that occurred was part of the A^2ATD. The purpose of the A^2ATD experiment was to develop and demonstrate a verified, validated, simulation within a DIS environment. AWC provided the Army with a capability to support this activity. The cell consists of three primary components: the Longbow Player Station (LPS), the CPS, and the Cell Manager (CM). LPS is a single crew station AH-64D Longbow Apache simulator developed by McDonnell Douglas, Mesa, Arizona. CM control is provided through the Integrated Threat Environment Management System (ITEMS™) simulation developed by CAE-Electronics, Montreal, Canada.

All three components of the AWC are DIS compliant and are networked together on a local DSI network. The AWC provides long haul participation on the world-wide DSI network. The AWC facility diagram is

Copyright © 1997 by the American Institute of Aeronautics and Astronautics, Inc. All rights reserved.

illustrated in figure 1.0

This paper describes the CPS component of the AWC. CPS is a dual crew simulator developed by Sikorsky Aircraft, Stratford, Connecticut, under the Boeing-Sikorsky Comanche contract.

4.0 COMANCHE PLAYER STATION

In February of 1994, the Army tasked Boeing-Sikorsky to determine the feasibility of providing a Comanche component to the AWC. A team of engineers, under the guidance of the Comanche Program Office evaluated existing government furnished equipment (GFE) for use in the CPS project. The basic objective was to provide a functional hardware, and software representation of what existed on the EDS at Sikorsky facilities in Stratford. The primary hardware task was to build a replica of the EDS. The software task involved the porting of the software load, including the air vehicle and mission equipment package, to a new host computer environment. The EDS software is primarily FORTRAN and C based code running on a cluster of Encore 87 and RSX series host computers. The new host computer system for CPS was the Harris Nighthawk™ UNIX based platform.

Development of CPS was accomplished in five major phases. These included:

Phase I established the requirements for the AvTB facility to support the CPS. This was accomplished through a site analysis of the AvTB that defined additional facility support items that would be required. During Phase I, an assessment was made to determine the level of usability of the equipment being provided as GFE. This hardware was previously purchased under the Rotary Wing Aircraft (RWA) Program, and was available for use in developing the CPS. The devices under evaluation were a set of Harris Nighthawk™ 4000 series host computers, and a set of ESIG 2000 image generators. A study was performed to evaluate the capabilities of the existing equipment against the requirements for the CPS. Findings and recommendations of the feasibility study and the front end analysis were presented at a preliminary design review (PDR).

Phase II included the design of the system architecture for the hardware, software, visual system, and a new real-time operating system. During this phase, the detailed interface control documentation was created that defined the major simulation components of the CPS. In addition, this phase also covered the major task of code transfer from the Comanche EDS Encore computers to the GFE Harris Nighthawk™ computer. The creation of the visual image software and the integration of the Evans and Sutherland image generator hardware was also completed. The development of a real-time executive and world interface software was created. Detail design of the hardware and software architecture were then presented at a critical design review at the completion of Phase II.

Phase III involved verification and validation of the simulation, initial acceptance test procedures (ATP), and local DIS network testing. The V&V test plan was developed and reviewed with the government during prior to the start of testing.

Phase IV involved the physical transfer of the fully integrated and tested CPS system from Stratford, Connecticut to the AvTB at Fort Rucker Site integration and a final ATP were performed. Once the CPS was integrated as a standalone device, a link was made with the LPS and the CM in a combined network test.

Phase V involved training personnel at the AvTB in CPS operations.

4.1 CPS Hardware

The major CPS hardware components consist of simulation computers, dual crewstation cockpit, visual display system, an operator station, and support equipment. The CPS system architecture is illustrated in figure 2.0

4.1.1 Simulation Computers

The primary computer system for the CPS is a single cabinet Harris Nighthawk 5806 UNIX based computer system. This system has six Motorola 88110 series RISC processors operating at 50 MHz and is known as the Simulation Host Computer (SHC). The SHC executes the simulation software modules synchronously, asynchronously, and in real-time on the six available processors. Top level software modules include the Comanche air-vehicle, mission equipment, world interface, I/O software, and the executive

Copyright © 1997 by the American Institute of Aeronautics and Astronautics, Inc. All rights reserved.

software. Peripheral hardware and software include: two Ethernet interfaces, SCRAMNet™ interface, NH-5800 runtime software, NH-5800 development software, HF77 FORTRAN compiler, and a 'C' compiler.

The next most significant computer system is the Image Generator (IG). The IG is an Evans and Sutherland 3000 Series system (ESIG 3000). This computer provides the CPS with the real-time visual environment. The CPS IG was customized to provide two channel processors for the forward visual scene known as the Out-the-Window (OTW), and one channel that functions as the aircraft sensor channel. The sensor channel generates the video output that represents the simulated Electro-Optical Targeting Display System (EOTADS) for the aircraft. The IG is capable of generating visual databases created from standard Digital Terrain Elevation Data (DTED), and Digital Feature Analysis Data (DFAD) sources. This allows the CPS to perform experiments any region of the world for which data is available. The IG also provides line-of-sight (LOS) and height above terrain (HAT) data that allows the interconnectivity between simulation devices in the experiment to be calculated and controlled. The IG generates the 3D image models which visually represent the other simulation elements that interact with the CPS. These 3D images include other simulated vehicles (moving and fixed type), and special effects such as explosions, missile fly-outs, weather effects, and time of day. The IG hardware is capable of generating up to 7000 polygons on the OTW display at a 60 Hz update rate. The Night Vision Pilotage System (NVPS) sensor and the EOTADS sensor are also updated at 60 Hz.

The Display Generators (DG) are a cluster of another six computer systems that are networked with the SHC and the IG. The DG generates the graphics required to drive the cockpit heads down and heads up displays. These computers are Silicon Graphics Incorporated (SGI) UNIX based graphics workstations. A variety of SGI computers with varying computational powers are used as elements of the DG. Individual system processing capabilities are determined by the complexity of the display supported. The Comanche cockpit consists of a System Management Display (SMD), a Tactical Situation Display (TSD) and a Helmet Mounted Display (HMD) system. The following table identifies the DG computer hardware used to support each particular display.

TABLE 1 - Cockpit Display Computers

Forward Seat Display	System
SMD	SGI Onyx Deskside RE II
TSD	SGI Onyx Deskside RE II
HMD	SGI-VGX 420

Aft Seat Display	System
SMD	SGI-Skywriter
TSD	SGI Onyx Deskside RE II
HMD	SGI-VGX 420

In addition to the systems illustrated above, a SGI INDY workstation performs an unique Comanche function known as target review. Comanche has the ability to rapidly perform a digital snapshot of the battlefield for later review by the pilots in a masked unexposed location. The INDY emulates this function by taking a digital snapshot from the CPS EOTADS sensor.

The SHC, IG, and DG are networked together by a fiber optic network called SCRAMNet™. This network operates at 150 megabits per second and provides a memory region that is virtually shared among unlike computer systems. A secondary Ethernet network is also available.

4.1.2 CPS Cockpit

The CPS cockpit is a functional replica of the Comanche EDS. Recall that the EDS is a primary design tool for the evaluation of the Comanche crew systems, handling qualities, and mission equipment subsystems. The CPS cockpit consists of a forward and an aft crewstation. Configuration of the CPS forward and aft crewstations are identical. This allows for redundancy and backup operation of all systems on the in the aircraft. Each crewstation consists of a center console, side console, sidearm controller, collective controller, and a helmet mounted display system.

The center and side consoles provide the pilot with all controls and displays required to operate the aircraft. The center console consists of two multi-function displays for aircraft system management and tactical situation data and two multi-purpose displays for tactical management indication. Illuminated switch panels for warning caution advisory, automatic flight controls, and weapons selection are also included. An

Copyright © 1997 by the American Institute of Aeronautics and Astronautics, Inc. All rights reserved.

interactive keyboard is also available for the pilot to input and receive digital communications, and adjust cockpit instrumentation settings. The side console consists of multiple switch control panels that provide for mission mode selection, engine control, jettison control, radio communication, HMD control, and cockpit lighting.

A Lear Astronics electronic sidearm controller provides primary flight control inputs. With the sidearm controller, the pilot is able to control the three primary control axes for flight, roll, pitch, and yaw. A linear collective control stick, located on the left side of the pilot, provides primary collective control. Both the sidearm and collective controllers have custom control grips with several switches used to operate multiple functions on the aircraft.

A binocular helmet mounted display (HMD) system provides heads-up displays for the pilot. The HMD provides the pilot with mission and aircraft critical symbology during daytime operations. The same symbology combined with night vision imagery acquired by the aircraft's NVPS system is used for night operations. The HMD is a Kaiser Wide-Eye™ 40° system that is configured with a 50% overlap. This yields an active instantaneous field of view of 40° vertical x 60° horizontal. The HMD has an Polhemus Fastrak™ electromagnetic head-tracker that provides head angles for the targeting function.

4.1.3 Out the Window (OTW) Display System

The planned OTW system consists of an 18 foot half dome with a Seos PRODAS™ S600 projection system. Field of view is 150° horizontal x 85° vertical. This visual system was originally procured for the Rotary Wing Aircraft (RWA) program and will be used for the CPS. The dome is presently under refurbishment and the CPS OTW display system is presently in an interim configuration. The interim system consists of three 37" Mitsubishi computer monitors located in front of the CPS cockpit. This provides a 120° horizontal x 25° vertical field of view.

4.1.4 CPS Operator Station

The CPS Operator Station (COS) is where the operator initiates start up, oversight, and shutdown of the CPS. A Graphical User Interface (GUI) called SPEAR was developed by Sikorsky Aircraft, to provide this control. SPEAR runs on an SGI Indigo workstation within the COS. The COS also contains a bank of video monitors that can easily display and record the crew member activity in the cockpit. The over-the-shoulder cameras and cockpit display repeaters allow the operator, test director, and subject matter experts to observe what is occurring in the CPS cockpit without disturbing the crew member during an experiment.

4.1.5 Audio and Video Distribution

CPS cockpit and operator station are linked through an analog communications system that replicates the intercommunications system and radio functions of the aircraft. The CPS has the ability to convert audio signals to a digital DIS format for transmission on the DSI network, permitting the crew members to communicate with other players on the digital battlefield. An SGI INDY workstation generates audio tones for radar warning tones and envelope limit cues.

CPS has an active video routing and distribution network that provides gain and frequency equalized video to all displays. The video distribution system dynamically switches RGB and component video to the cockpit video displays and the HMD system. Operating video standards employed are RS-170 and RS-343A. The video system also performs active luminance keying required for mixing symbology and FLIR in the HMD and on the heads down displays for targeting. This system is under software control and operates interactively with pilot inputs from the cockpit controls.

4.1.6 DIS Interface

The CPS is DIS compliant to the 2.0.3+ and 2.0.4 versions and can interact with other DIS compliant devices on the DSI network. Physical interface to the DSI network is Ethernet. The CPS has two DIS ports. The primary port is the world interface to the CPS simulation, while the secondary port is the digital audio interface to other players on the network.

4.2 CPS Software

Software currently executing on the CPS was derived from a configuration managed (CM) load executing on the Sikorsky Comanche EDS simulator at Stratford.

The primary task for CPS development involved a

Copyright © 1997 by the American Institute of Aeronautics and Astronautics, Inc. All rights reserved.

migration effort of the CM load from the EDS simulator to the CPS platform. The CPS baseline software consists of several modules that can be categorized at a high level. These software subsystems included: the air-vehicle (AV), the mission equipment package (MEP), the pilot vehicle interface (PVI), the world interface (WI), the display generation software (DGS), the visual software (VS), the input /output (I/O), and the executive software. Approximately 56,000 lines of host code, and 26,000 lines of DGS code are in the baseline CPS load. Execution rates of the various software modules range from 1 Hz to 120 Hz. The overall architecture is modular and is illustrated in figure 3.0.

4.2.1 Air Vehicle Model

The AV software is a complex series of algorithms that enable the Comanche simulation to accurately represent the aircraft in terms of handling qualities and flight controls. This software was originally derived from the Sikorsky advanced Generic Helicopter (GENHEL) libraries. The GENHEL concept allows a simulation developer to tailor, from library based modules, an aircraft simulation to represent a specific aircraft. The AV model consists of a flight control system, equations of motion, airframe aerodynamic characteristics (including the fuselage, empennage, and weapons bay doors), engines, drive train, rotors, swash plate, and environmental effects. Actual airframe design data are programmed into the GENHEL model and an accurate representation of Comanche is produced. The model includes Comanche specific data for the LHTEC T800 engines, and the FANTAIL™ anti-torque system. The AV model incorporates the Comanche fly-by-wire flight control system including the automatic flight control system (AFCS) and primary flight control system (PFCS).

4.2.2 Mission Equipment Package Model

MEP simulation software consists primarily of the weapons system, mission system, navigation system and EOTADS system software. CPS MEP modules consists of a six degree of freedom infrared Hellfire missile system, 20 millimeter turreted gun system, 2.75 Hydra-70 rocket system, six degree of freedom Stinger missile system, and a comprehensive stores management system. The EOTADS system includes an aided target detection and classification system (ATD/C). ATD/C utilizes the turret mounted FLIR and Day TV to acquire and classify targets. The system also has the ability to simultaneously track targets with the automatic target tracking (ATT) model. Finally, the CPS EOTADS simulation has a function known as Bulk Data Memory (BDM). This function enables the pilot to scan the battlefield, take a digital snapshot, and review the threats while remaining masked.

The mission system software presents the pilot with mission time and other mission critical parameters. Aircraft system health monitoring is also performed by this subsystem. The MEP navigation software consists of several modules that support the navigational functions such as flight condition presentations to the pilot on the heads down and HMD displays. CPS MEP also simulates the flight path management system which includes the insertion and deletion of tactical waypoints on a digital map located on the TSD.

4.2.3 Pilot Vehicle Interface

PVI software simulates the man-machine interface through which the pilot and copilot interact with the entire MEP system. The PVI system software controls the MEP displays and menu systems presented to the pilots.

PVI displays symbol sets on the HMD, digital and map overlays on the TSD, the aircraft health pages and flight instrumentation on the SMD, and the threat management information on the multi-purpose displays (MPD). The PVI software also simulates the Comanche's menu systems. The PVI manages several levels of menus for each major sub-system that is simulated. These menu structures allow the crew members to configure and control the target acquisition system and weapons system, digital map, HMD symbology presentation, and NVPS.

4.2.4 World Interface

World Interface software provides the mechanism to connect the CPS to the DSI network. This bi-directional interface allows the CPS to interactively participate on the digital battlefield. This software transmits and receives entity states of the CPS ownship as well as other players participating in the exercise. WI software also manages the weapons systems outgoing and incoming data regarding firing, detonation, and probability of hit and kill data. The WI accepts information from the DSI network and is

Copyright © 1997 by the American Institute of Aeronautics and Astronautics, Inc. All rights reserved.

responsible for the remote player management task on the CPS environment.

CPS supports the DIS message sets that allow digital communications between CPS, LPS, CM, and an Army Airborne Command and Control (A^2C^2) station. These message sets are known as the Interoperability Data Message (IDM) messages. These were developed for the AWC in support of the A^2ATD Experiment #5. The CPS also has the capability to generate encoded audio communications over the DSI network, and support radio communications to remote players on the digital battlefield.

4.2.5 Display Generation Software

DGS is ANSI C code that executes on Silicon Graphics workstations. The DGS software modules generates the cockpit video displays to support the PVI, such as the: HMD symbology, TSD, SMD, and TAS. All of these modules are linked via a reflective type memory system to the SHC, and operate at update rates ranging from 20 Hz to 60 Hz.

4.2.6 Visual Software

The visual software (VS) was developed to provide an interface from the simulation host computer to the IG. This software manages the control of the IG and its visual effects. VS provides control of moving models and line-of-sight (LOS) calculations.

The IG is capable of supporting both day and night operations. During day operations the IG is configured to drive an OTW visual display and support the TAS sensor channel. During night operation, the system provides the Comanche NVPS as well as TAS sensor.

4.2.7 Input/Output (I/O) Software

The I/O system software consists of device drivers that support the various cockpit hardware systems on the CPS. These systems interface the cockpit control panels, sidearm controllers, collective controllers, and control grip functions. There are also several cockpit devices that require network interface and serial communications.

4.2.8 Executive Software

The executive software provides the means of control of the CPS simulation through the previously described SPEAR. The CPS operator can control various functions of the CPS simulation from the SPEAR utility. Functions available include simulation run, freeze, re-initialize, and abort. This software utility is available to the operator at the COS station and allows him to monitor certain remote player statistics in relation to the CPS as well as permitting him to observe data with respect to threat locations on the battlefield. SPEAR also gives the operator a software symbol lookup capability for status, debugging, and maintenance.

5.0 USES OF THE CPS

The initial delivery of the CPS was originally planned for the end of 1995. There was a program decision to bring the device to the AUSA annual symposium in Washington, DC. This event occurred in October 1995. After that demonstration the device was permanently installed in the AWC at the Aviation Test Bed at Fort Rucker, Alabama. The first experiment that the CPS participated in was the A^2ATD, Experiment #5 in February 1996. Since the conclusion of that experiment, the CPS has been used for pilot familiarization and development of Comanche TTP's by pilots of the Army's Comanche EOC team. A brief description of the October AUSA Demonstration (OAD'95) and the A^2ATD Experiment #5 illustrates CPS utilization.

5.1 OAD'95

The purpose of OAD'95 was to demonstrate a Synthetic Theater of War (STOW). It was sponsored by the Army's Louisiana Maneuvers (LAM) Task Force. This was accomplished through the use of simulators located worldwide that were linked together by the DSI network. The goal of STOW was to evaluate the use of simulation and simulators as tools for combat development, materiel development, system acquisition, test, and evaluation. By collecting data on the performance of a validated simulation system, a design team can refine the design of an evolving system long before the hardware prototype is built. Feedback from exercises like OAD'95 allow the system designers to maximize the effectiveness of the designs.

The CPS represented the Comanche in the STOW exercises of OAD'95. There were three scenarios in

Copyright © 1997 by the American Institute of Aeronautics and Astronautics, Inc. All rights reserved.

which the CPS participated. These included: the deep strike, close battle, and the dismounted infantry scenarios. These experiments demonstrated the capabilities of the Army Theater Missile Defense Element (ATMDE) as well as the ability of Comanche to plan and execute the Theater Missile Defense (TMD) and deep attack operations.

5.2 A²ATD Experiment #5

A²ATD was a DIS exercise that was sponsored by the Army Materiel System Analysis Activity (AMSAA). The purpose of this activity was to develop and demonstrate verified and validated (V&V) simulations that emulate anti-armor weapons systems. Several experiments were included in the A²ATD, which were designed to utilize the synthetic environment created by the Battlefield Distributed Simulation-Developmental (BDS-D) simulation initiative. Experiment #5 involved the use of the CPS in armed reconnaissance and deep attack scenarios. The mission scenarios were designed to further the development of TTP between Comanches and Longbow Apaches on the digital battlefield. The Comanche crew and the Apache Longbow crew communicated digitally on the battlefield while performing true cooperative missions.

This exercise provided system designers of Comanche valuable insight to the potential effectiveness of the combined weapons system. The objectives of Experiment #5 were successfully met. Today, a validated DIS compliant Comanche simulation capability supports the Army aviation community.

5.3 Future Uses

Since the A²ATD experiment there has been a high degree of interest to include Comanche in other digital battlefield experiments. Force Multiplier 2000 and Bird-Dog are two potential initiatives that involve Comanche and Unmanned Air Vehicles (UAV). The Comanche program plans to use the CPS for continued crew familiarization and further development of TTP's. In addition, the Program intends to use the device for the Force Development Test and Evaluation (FDTE) phase of the Comanche development test program.

As part of the Comanche development program, the CPS will be upgraded and maintained in accordance with program design updates.

6.0 CONCLUSION

CPS is an important asset for the Comanche program. The simulator is a tool that the Army can use to perform evaluations of the Comanche system design long before the aircraft enters production. The use of simulation is increasing in importance, especially since the initial operational capability (IOC) aircraft will not be fielded for several years. Simulation will continue as a tool for requirements definition, materiel development, system acquisition, test, evaluation, and training. Simulation provides a cost effective way to demonstrate and evaluate new weapons system before spending billions of dollars on development and refinement of prototype hardware.

CPS has given Army Aviation a preview of what to expect from Comanche when it enters the field in the 21st century. High fidelity simulation and digital connectivity with other systems over the DSI network better prepares troops for combat in the future. The decrease in cost to performance ratios of computers and simulation technology have allowed the fielding of low cost simulators that enhance troop readiness. This trend will continue in the future.

Sikorsky and the Comanche Program office have developed a new Comanche simulator since the completion of the CPS. This device is based on the CPS and is known as the Comanche Portable Cockpit (CPC). The CPC is a mobilized version of the CPS. The primary purpose of the CPC is to increase awareness of Comanche and demonstrate Comanche capability to the armed services. The device is highly mobile and available for world-wide transport and operation. The addition of the CPC provides another manned virtual simulator to the BDS-D environment. CPC is currently touring the United States.

7.0 ACKNOWLEDGMENT

The author would like to acknowledge the dedication and commitment of the Simulation organization at Sikorsky Aircraft in Stratford, Connecticut. Their creativity, determination, and flexibility applied to the CPS project was instrumental in its success.

Copyright © 1997 by the American Institute of Aeronautics and Astronautics, Inc. All rights reserved.

Figure 1.0 Aviation War-fighting Cell (AWC)

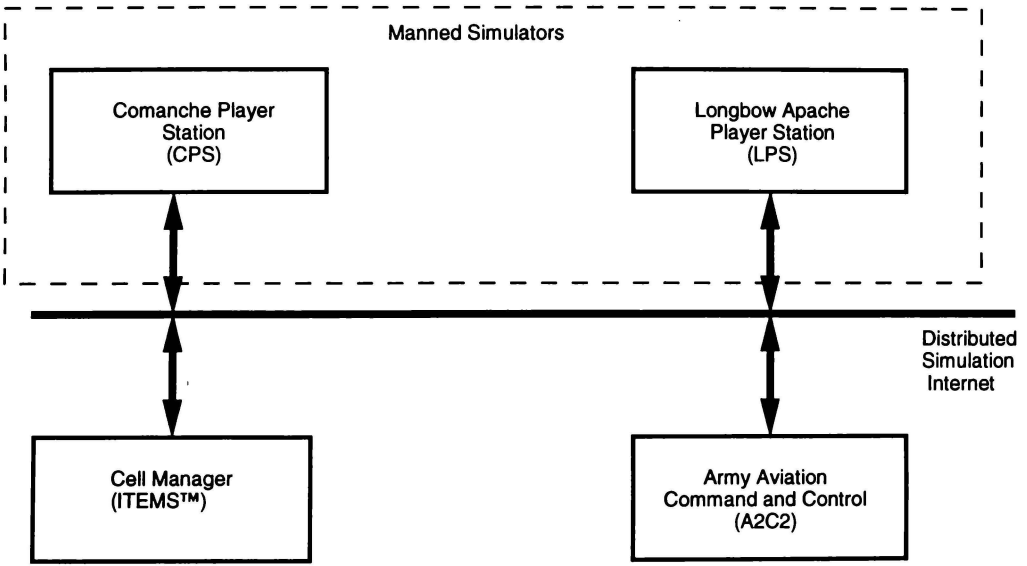


Figure 2.0 CPS System Architecture

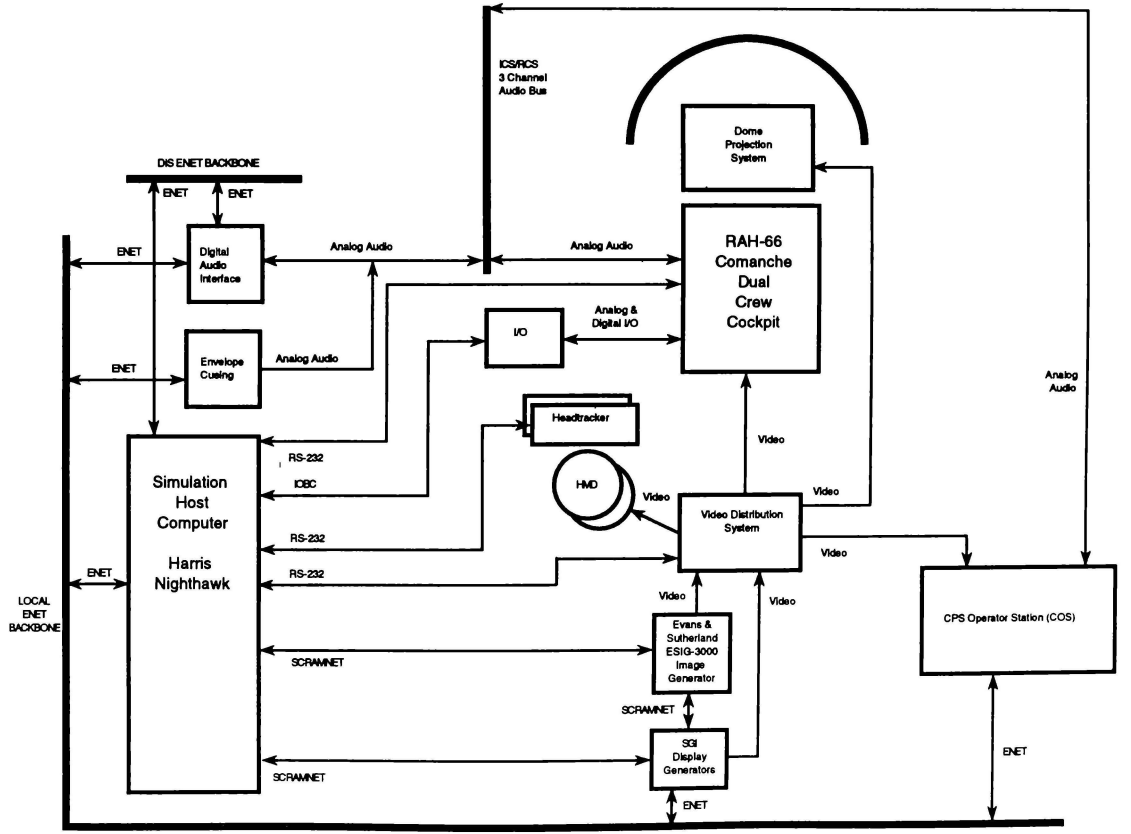
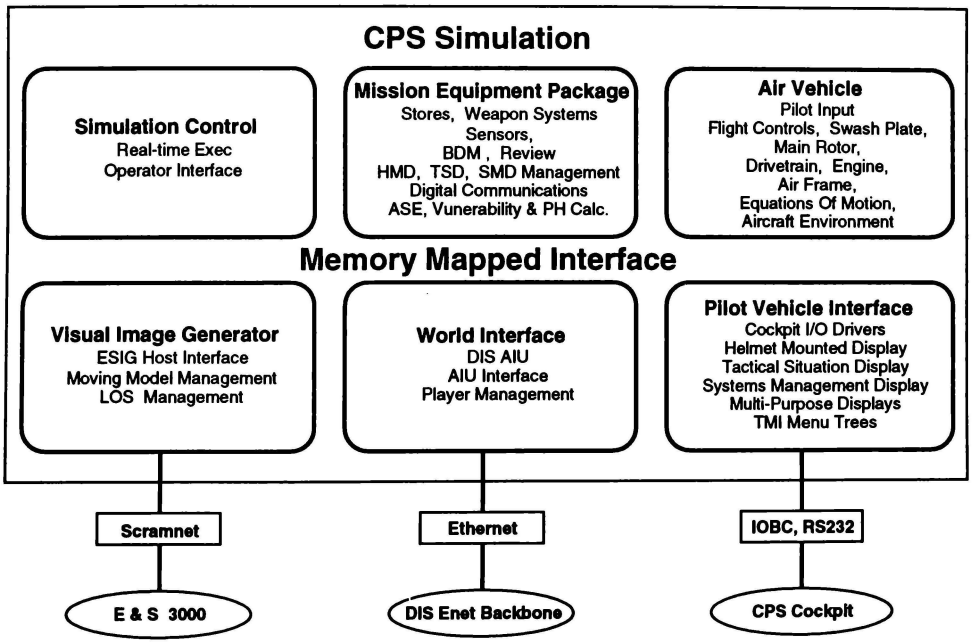


Figure 3.0 Software System Architecture



MODELING AND SIMULATION FOR SMALL AUTONOMOUS HELICOPTER DEVELOPMENT

Eric N. Johnson*
work conducted employed at
The Charles Stark Draper Laboratory, Inc.
Cambridge, Massachusetts USA
now employed at
Lockheed Martin Aeronautical Systems
Marietta, Georgia USA

Paul A. DeBitetto*
The Charles Stark Draper Laboratory, Inc.
Cambridge, Massachusetts USA

Abstract

The Charles Stark Draper Laboratory, Inc., Massachusetts Institute of Technology, and The Boston University have cooperated to develop an Autonomous Aerial Vehicle (AAV) that competed in and won the 1996 International Aerial Robotics Competition, sponsored by the Association for Unmanned Vehicle Systems, International (AUVSI). Development of the vehicle continues to support on-going research in the area of autonomous systems. A simulation capability has been developed to support the design, development, and test of the navigation, control, guidance, and vision processing sub-systems, as well as human-machine interfaces and procedures. The use of the simulation described in this paper is identified as a key factor in the success of the program at the competition and operations since.

Introduction

The International Aerial Robotics Competition, organized by the Association for Unmanned Vehicle Systems, International (AUVSI), provides a unique opportunity to develop an autonomous vehicle system with many of the same features, components, and potential pitfalls as fielded unmanned vehicle systems - and to test that system in a real-life/competition environment. A team was formed between students, faculty, and staff of the Massachusetts Institute of Technology, Boston University, and the Charles Stark Draper Laboratory, Inc. that competed in and won the 1996 competition, held in Orlando, Florida.

The competition consists of a 60 by 120 ft. field with five randomly placed drums [1]. Each drum has one of two kinds of labels. A contestant's vehicle must start and finish in a 15 by 15 ft. area in a corner of the field, provide a map of the location and classification of the drums, and pick up a small disk on one of the drums. During a contest attempt, the vehicle and any ground equipment must operate completely autonomously, i.e. no human operators.

The Draper small Autonomous Aerial Vehicle (AAV) system consists of an aerial vehicle (Figure 1), a Ground Control Station (GCS), a vision processor, and a safety pilot. Figure 2 is a functional block diagram and shows how the aerial vehicle, GCS, and vision processor fit together. The aerial vehicle performs navigation and control on-board using a redundant suite of sensors including: a Differential GPS (DGPS) unit, an Inertial Measurement Unit (IMU), a sonar altimeter, and a flux compass. Guidance and operator control happens on the GCS. The helicopter also carries a camera and transmitter that provides real-time video images to the ground. A ground based vision processor then converts the image data into drum position and classification estimates.

The contest field size requirements, the restricted take-off and landing area, a desire for flight characteristics that allow manual control, and the payload capacity desired led to the selection of a helicopter for the aerial vehicle. The aerial vehicle is an off-the-shelf radio-controlled helicopter with a six foot rotor diameter produced by TSK, called the Black Star. The helicopter has an empty weight of 15 lb. and a payload capacity exceeding 9 lb. The helicopter is powered by a 32 cc gasoline engine.

On-board Processing

A PC-104 stack is used as the main processing module for the vehicle. In particular, an Ampro 486 DX2 processor is used. The navigation and control algorithms run on-board.

* Member AIAA

Copyright © 1997 by the American Institute of Aeronautics and Astronautics, Inc. All rights reserved.

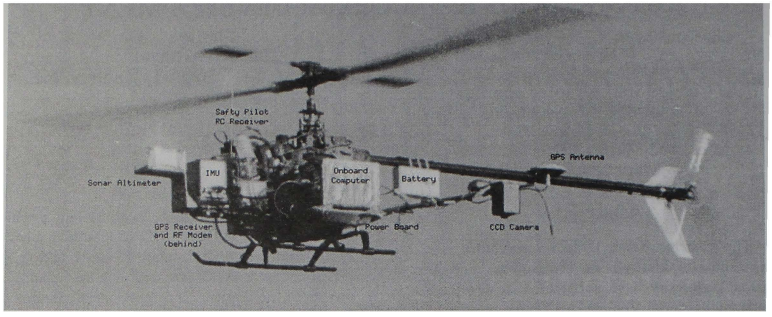


Figure 1. 1996 Draper Small AAV

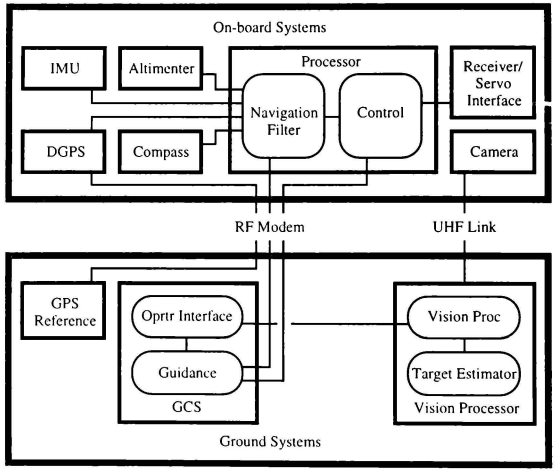


Figure 2. System Functional Block Diagram

Navigation

Navigation Sensors

A DGPS system with local ground reference station is employed to supply low bandwidth position and velocity measurements to a navigation filter. The NovAtel RT-20 system was used.

An IMU was employed to provide high bandwidth angular rates and linear acceleration measurements to the navigation filter for updating the attitude, angular rates, linear position and velocity state estimates. The sensor used is the Systron Donner Motion Pak, consisting of three quartz rate gyros and three quartz flexure accelerometers.

The sonar altimeter provides high rate/resolution altitude information at close range. A flux compass is utilized to prevent long-term drift of the navigation filter heading estimate.

Navigation Algorithm

The navigation filter merges the sensor information from the IMU, DGPS unit, sonar, and compass. Because these sensors have unique update rates and error properties, an extended Kalman filter was selected to accomplish the task. When updates of the DGPS, sonar, or altimeter occur, an optimal discrete update [2] is made of the state vector and the state error covariance matrix.

Guidance and Control

Receiver/Servo Interface

An on-board receiver/servo interface allows the safety pilot to select between *ground transmitter controlled* and *on-board processor controlled* modes. The interface is powered by the R/C receiver battery (independent of the power supply for the on-board processor, DGPS, other sensors, and modem), and communicates with the on-board computer via a standard serial port.

Control Law

The control system is designed to generate effector commands for the roll and pitch cyclic, collective pitch, tail rotor pitch, and throttle based on a commanded position, heading, and velocity. It utilizes feedback from the navigation system, including position, velocity, attitude, and angular rate estimates. The ability to command a velocity in addition to position and heading allows the control system to eliminate steady state errors during forward flight or other type of commanded movement.

The control system is divided into four loops: roll, pitch, yaw, and collective/throttle. The loops are used to control lateral position, longitudinal position, heading, and altitude respectively. The trim positions of cyclic, collective, and tail rotor are used as internal states, this allowing physical interpretation of these states and allowing the safety-pilot to pre-trim the helicopter to minimize initial transients in the system.

Guidance

The guidance system generates position, heading, and velocity commands based on the current state of the helicopter as reported by the navigation system and a waypoint list. The guidance system commands a straight line course between waypoints at a nominal velocity, and transitions between these trajectories using a nominal acceleration. The values used for the nominal acceleration and velocity can be changed by an operator at the ground station.

Communication

The radio modem link performs three main functions. First, it sends navigation information from the helicopter to the ground computer. Second, it relays control system commands from the ground computer to the helicopter. Finally it is used to send differential GPS data from the ground GPS receiver to the on board GPS receiver.

Ground Control Station (GCS)

The GCS including the operator interface utilizes an Intel 486-based laptop. During autonomous flight, displays shown on the GCS allow monitoring of the navigation system estimates, state of the communication link to the helicopter, waypoint list, DGPS solution status, state of the control system, and state of the vision processing. Also, the interface is used to perform other operations.

The display consists of four elements. The primary flight display, depicted in Figure 3 depicts state information such as position, velocity, and attitude as reported by the navigation system. The navigation display is an electronic map of the flight area including the helicopter position as reported by the navigation system, the location of the DGPS reference station, the waypoint list, and the commanded position given the control system. The annunciator panel shows the status of most sub-systems. The annunciator for any particular system can take on various colors and text labels depending on the state the system. The systems pages provide access to the details of specific sub-systems.

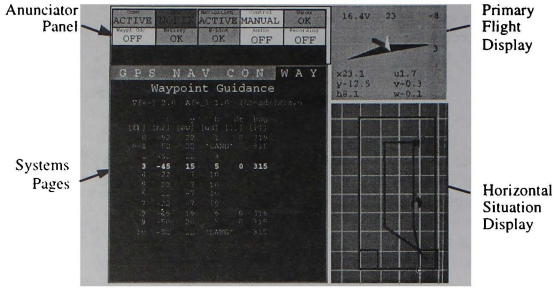


Figure 3. Operator Interface Displays

Vision Processing

A greyscale CCD camera with a UHF transmitter was mounted on the helicopter. It was downward facing, so drum labels will be visible. Contest vision processing is broken into three steps: Drum location, drum classification, and target estimation.

The drum detector is composed of three parts: a gradient operator, matched filter, and a peak extractor. The matched filter processes the gradient image. Since the labels characteristically have many edges, they are easy to pick out against the relatively smooth background in the gradient image by matching to regions of clustered edges. The expected size of the labels is calculated from the height of the helicopter and this determines the size of the matched filter template. The matched filter is also designed to be insensitive to a long edges such as those from the rim and sides of the drum. Then the peaks of the matched filter output are the most likely positions of the labels, and hence the drum positions.

The label classifier matches the pixels around the label position estimates to templates. The orientation of the input pixels is normalized to the vertical, so only two templates are necessary for each label type (The second template is just the first rotated by 180 degrees). The scores of the template match affect the confidence that the contact is an actual drum.

An optimal estimator combined outputs from the vision processor along with its reported confidences into a single list of probable drum locations and classifications. The five solutions with the highest confidence were turned into the judges as the system output.

Simulation Capability

A simulation capability was put in place to support the design, development, and test of the navigation, control, guidance, and vision processing sub-systems, as well as man-machine interfaces and procedures. Also, it was used for safety-pilot training. The simulation capability includes:

modeling:

- Main and tail rotor forces and moments, including tip-path dynamics of the main rotor and Bell/Hiller flybar arrangement.
- Aerodynamic forces and moments due to the fuselage and tail surfaces
- Ground contact forces and moments of main gear and tail
- Sensor models of DGPS, IMU, and sonar altimeter
- Servo dynamics

hardware/software compatibility:

- Hardware compatibility with receiver/servo interface, allowing piloted flight using actual safety-pilot transmitter
- Hardware compatibility with ground computer (GCS), allowing tests of all man-machine interfaces
- Software compatibility with on-board processing, allowing the same software to be used in the simulation as used on board the actual AAV

scene generation and analysis:

- Scene generation including contest elements, such as the field and drums with labels
- Parameter logging, un-logging, and plotting
- Linear model extraction
- Time control: Fast time, slow time, pause, step, and real time

This simulation capability has played a key role in the success of Draper small AAVs to date, and is described in the following sections. The ways in which this simulation capability was utilized is also described in following sections.

Modeling for Simulation

Dynamic modeling for this project included a 6-degree-of-freedom rigid-body representation of the motion of the helicopter. In addition, main rotor and main rotor flybar first order flapping dynamics are included. The main and tail rotor forces and moments are determined using an actuator disk theory formulation. Also, a ground contact model was implemented. This model was based largely on the material in reference [3], with important extensions described in this section.

Rotor Thrust and Power

Since the main rotor tip speed of the TSK Black Star is approximately 400 feet per second, and the largest dimension of the contest arena is 120 ft, the maximum expected tip speed ratio is less than 0.05. Asymmetric wake and retreating blade stall can be neglected in rotor modeling for this range of tip speed ratios, dramatically simplifying the calculations necessary for rotor thrust and power.

Using the formulation for a perfect actuator disk in an oblique flow, the following two equations must be solved simultaneously to get the main rotor thrust and induced velocity [3] and [4], this is accomplished numerically:

$$T = \rho \frac{abc}{4}\left(w_b - v_i\right)\Omega R^2$$

$$v_i = \frac{\eta_{GE} T}{2\pi R^2 \rho \sqrt{u^2 + v^2 + \left(w_r - v_i\right)^2}}$$

where:

T	Main rotor thrust
v_i	Main rotor induced velocity at the rotor

and:

a	2-D lift curve slope of rotor blade
b	Number of blades (2)
c	Chord of rotor blades
R	Radius of rotor blades
Ω	Angular rate of main rotor rotation
u, v, w	Velocity of the rotor with respect to the air expressed in the body frame, u forward, v to the right, and w down
ρ	Air density
w_r	The velocity of the rotor with respect to the air, normal to the disk plane; it can be calculated by: $w_r = w + a_1 u - b_1 v$
w_b	The velocity of the average blade with respect to the air, normal to blade; it can be calculated by: $w_b = w_r + \frac{2}{3}\Omega R\theta$
η_{GE}	Reduction in induced velocity due to ground effect; it is calculated assuming that ground effect is of the form [4]: $\eta_{GE} = \frac{K_{GE}}{1 + \left(\frac{2R}{h}\right)^2}$

where:

a_1, b_1	Main rotor first order flapping, pitch and roll
θ	Main rotor pitch (collective pitch angle)
K_{GE}	An empirically determined value, setting the magnitude of ground effect

h	The height of the main rotor above the ground

Once rotor thrust and induced velocity are determined, induced and parasite power can be calculated with [3] (parasite power) and [4] (induced power):

$$P_{Induced} = T\left(v_i - w_r\right)$$

$$P_{Parasite} = \frac{\rho}{8} c_{Do} R^2 bc\Omega\left[\Omega^2 R^2 + 4.6\left(u^2 + v^2\right)\right]$$

where:

$P_{Induced}$	Induced power of the main rotor
$P_{Parasite}$	Parasite power of the main rotor
c_{Do}	2-D zero-lift drag coefficient of a rotor blade

The tail rotor thrust, power, and torque are calculated in the same manner as the main rotor. The most significant difference is that ground effect is neglected.

Main Rotor and Flybar Flapping

As described above, first order flapping dynamics are included for the main rotor. In addition, the TSK Black Star has a Bell/Hiller flybar main rotor arrangement, as can be seen in Figure 1. This arrangement is common in small helicopters, and serves to stabilize the helicopter in roll and pitch - making manual control considerable easier.

The flybar adds dynamics to the system that need to be modeled for Guidance, Navigation, and Control (GN&C) analysis or training. The approach taken, discussed briefly here, was to use the first order flapping dynamics as developed for the main rotor in [3] and applying them to the flybar - as though it was another rotor disk. Once this is completed the mechanical connection between the swash plate and the flybar must be included as a gain between commanded main rotor flapping and commanded flybar flapping. Also, a gain between flybar flapping and commanded main rotor flapping must be included.

Ground Interaction

In order to simulate entire missions, take-offs and landings must be available in the simulator. To this end a ground interaction model was include, that is the forces and moments generated as the helicopter landing skids or tail boom hit the ground.

Contact points on the helicopter body (the skids and tail) were modeled as 3-D springs and dampers, with static and dynamic friction coefficients. Once the proper velocities and deflections have been calculated, the force in the vertical direction:

$$F_z = -K_d d - K_d w_p$$

for positive deflections, where:

d, w_p	Calculated vertical deflection and deflection rate of the ground contact point
F_z	Vertical force due to contact point
K_d, K_d	Spring stiffness and damping parameters

When a contact point is not skidding with respect to the ground, a static friction coefficient is used, and a spring and damper generate forces and moments to keep the contact point were it is on the ground. This approach prevents the helicopter from moving around when small forces and moments are generated by the rotors while still firmly planted on the ground. Another consequence is that two additional states result for each ground contact point. The physical interpretation of these two states is the longitudinal and lateral deflection of the contact point with respect to the body of the helicopter.

To use the lateral deflection as an example, the side force due an particular contact point is proportional to this lateral deflection and the rate of change of lateral deflection. Prior to skid, the rate of change of the lateral deflection is the lateral velocity of the contact point with respect to the ground:

$$\dot{s}_y = -v_p$$

$$F_y = K_s s_y + K_s \dot{s}_y$$

where:

s_y	Lateral deflection of contact point
$\dot{s}_y$	Lateral deflection rate of contact point

v_p	Lateral velocity of the contact point with respect to the ground
F_y	Lateral force due to contact point
K_s, K_s	Spring stiffness and damping parameters

Once the static friction force is exceeded (proportional to the vertical force at the contact point), a more conventional ground interaction model is used, with a spring and damper force normal to the ground, and a friction force generated opposite the direction of the skid.

Sensor Models

Sensor models were included, particularly for navigation algorithm development. DGPS system errors are modeled as colored Gaussian noise. The IMU sensor model errors included: limits, scale factor, bias, and Gaussian noise. The compass and sonar models both also included limits, scale factors, biases, and Gaussian noise. All sensor model noise parameters were based on engine running static tests. In addition to error analysis, having sensor models that included all the same calibration, reference frame, and unit issues as the real sensors dramatically reduced the amount of software development that needed to be done using the real sensor hardware.

Analysis and Data Visualization

Scene Generation

A scene generation system was developed for Silicon Graphics, Inc. (SGI) workstations utilizing the OpenGL graphics language. The scene includes the contest arena, helicopters, drums, clouds, and trees. Piloted simulations revealed the importance of detailed textures and background (including trees and clouds) so that the human pilot can judge the motion of the helicopter relative to the background. A sample image is shown in Figure 4, depicting the view of the helicopter as seen from a 6 foot tall person standing just outside the contest volume.



Figure 4. Draper Small AAV Simulator Scene Generation

For the helicopter scene object, it was found to be quite useful for GN&C development and training to show the main rotor coning and first order flapping generated by the helicopter model as changes in geometry of the spinning disk drawn in the scene. The situation in analogous to depicting rudder and elevator deflections on the scene object for an airplane.

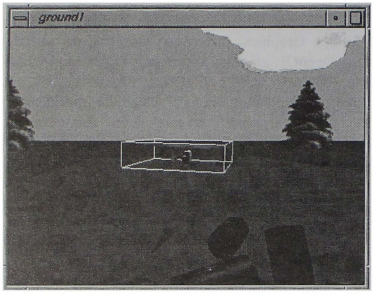


Figure 5. Navigation Solution Depicted in Scene

In addition, the depiction of the navigation solution was also depicted. This was a useful analysis tool for working with the navigation sensor fusion algorithms. A sample image is shown in Figure 5. If the navigation system is working properly, the white box (depicting the navigation solution) will contain the helicopter.

Plotting, Logging, and Un-Logging

The simulation utilized the CSIM Framework, a Draper Laboratory simulation environment which, among other things, includes utilities for real-time data plotting and logging. In addition, all simulation parameters can be examined and modified during execution. There is also a logged data playback feature.

Linear Model Extraction

A linear model extraction utility was also implemented, to facilitate linear analysis of the plant. It was implemented as a command that could be executed at any time. This command would linearize the plant, including the rotor flapping states, the rigid body states, the effectors, and the sensors at the current flight condition. The text file output was then imported into one of the common control system analysis packages.

Uses of Simulator

GN&C Development and Test

One of the most common uses of the simulator was to connect the GCS to the simulator running on an SGI workstation. This mode is suitable to test all Guidance, Navigation, and Control (GN&C) algorithms, most software, and GCS operator interface. This configuration is depicted in Figure 6.

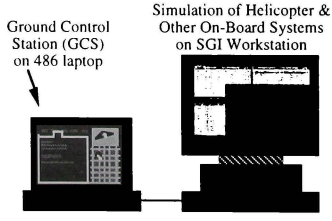


Figure 6. GCS Hardware-in-the-Loop Configuration

The software which normally runs on the on-board processor is also compiled into the simulation - allowing this software to be tested during the simulation. Also, this allowed GN&C development to occur using the same software which runs on the helicopter - avoiding the need to "port" the code to the new platform once the algorithms were developed or a change was made. This software included the control system, the navigation system, and the

communication protocol with the GCS and vision processing element.

This capability was used extensively prior to hardware completion and prior to flight test. In addition, anomalies discovered in flight test were, as standard practice, corrected in the simulation BEFORE a modification was tried in flight test. In addition, a linear model extraction utility was implemented and used to facilitate off-line linear analysis.

Software Development and Test

The next level of hardware-in-the-loop is to include the receiver/servo interface and the safety pilot transmitter. This allows the test of the safety-pilot interface, the receiver-servo interface hardware, and more software. Also, the simulation can be used for training all flight test operators in this configuration: the GCS operator and the safety pilot. This configuration is depicted in Figure 7.

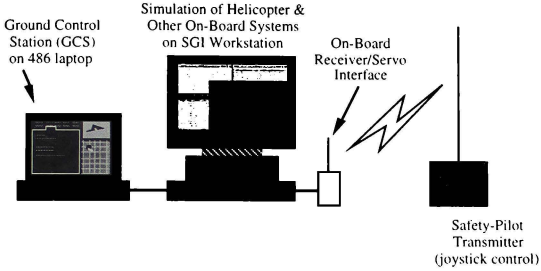


Figure 7. Hardware-in-the-Loop Configuration

The hardware-in-the-loop simulation configuration depicted in Figure 7 facilitates real-time testing of almost all software used with the vehicle (both on and off board). Software work performed outside the safety of the simulator was confined primarily to the low-level interfaces to the navigation sensors. In addition to the obvious benefit of testing the software, this also had the effect of making the software implementation job much less tedious. This was particularly true of the GCS software, where new functionality could be immediately tested in a flight test environment. This gave immediate feedback as the success or failure of a change.

Operator Interfaces and Procedures

The hardware-in-the-loop simulation configuration depicted in Figure 7 facilitates a detailed mission rehearsal, including both the GCS operator and safety-pilot. Tests can be performed repeatedly in the simulator before they are done for the first time in flight test. This allowed a myriad of operator interface design issues and test procedures to be ironed out prior to using up valuable flight test time.

Training

Because the hardware-in-the-loop simulation configuration depicted in Figure 7 facilitates a detailed mission rehearsal, including both the GCS operator and safety-pilot, it is also an effective training tool. This gives the GCS operator and safety-pilot the experience necessary to handle in-flight problems more effectively. In addition, it is an excellent environment to learn and practice normal and emergency procedures.

Image Processing Development and Test

The use of logged data playback allowed almost all vision processing development to take place prior to its initial flight test, depicted in Figure 8. Flight tested were performed while the navigation system solution and the on-board video image were recorded. The navigation data and video images were then played back and fed to the vision processing sub-system in real-time.

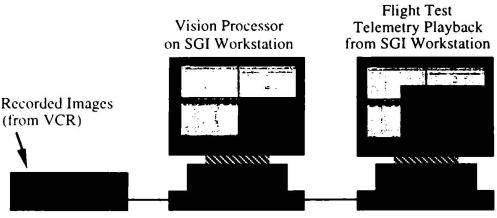


Figure 8. Vision Processing Test Configuration

In addition to the above method, which utilizes recorded video to generate realistic images to the vision processing system, another configuration was also used. Specifically, the scene generation system was set up to generate a scene that represented the vantage point of the on-board camera. Although testing image processing systems with synthetic images is not a rigorous test, since real-life degradation effects were not present, it was nonetheless a valuable test for checking algorithms and software for varied geometries, and under a more controlled set of conditions than available during flight tests. As a result, this was another useful form of testing.

Results and Conclusions

The MIT/Boston University/Draper lab entry won the 1996 International Aerial Robotics Competition. Seven flights were made. During the winning flight, all five drums were located, and two classified correctly. No attempt was made to pick up the disk. It was the first time in the history of that contest that a vehicle had flown to a series of waypoints completely autonomously take-off to landing, only touching the ground in the designated area. In addition, this AAV also executed a mission.

The following conclusions were drawn relating to the use of simulation for AAV development and test:

1) Simulation is an essential part of guidance, navigation, and control design and development. The extensive use of simulation for this purpose generated a design that only needed to be tuned in flight test, saving the program several weeks and perhaps a crash of the aerial vehicle. Also, using the simulation as a tool to correct problems uncovered in flight test saved considerable time as well.

2) Simulation is an essential part of software development. Developing interfaces, real-time code, and simply executing all possible code paths is greatly facilitated by a simulation capability. In addition, the act of software development itself is considerably less tedious, and efforts are more focused when additions and changes can be immediately tested. Also, it is believed that the use of simulation for software development saved the program several weeks by allowing software to developed concurrently with the associated hardware.

3) Simulation is an essential part of operator interface and development. Mission rehearsals led to many changes to the interfaces. A design without test in the simulation would have required changes during the flight test phase.

4) Utilizing logged telemetry data and recorded on-board video can be effectively used to develop and test vision processing schema. Tuning of the vision processing was done almost completely in the lab using this technique.

5) The maintaining of a well-trained crew to operate the system is critical during a flight test program. Human operators often need to make up for equipment failures or newly discovered problems. Hand signals, checklists, consistent terminology, and simulation training/mission rehearsal where effective tools.

In short, the extensive use of the simulation described in this paper was a key factor in the success of the program at the Aerial Robotics Competition and operations since. The close relationship between flight test and simulation allowed the flight validation of the guidance, navigation, and control sub-systems to take place within a two week period, clearly indicating that most development work was successfully completed in simulation. It allowed

most software problems to be addressed before flight test. It made software development less tedious, since additions could be utilized immediately in the simulator. Mission rehearsals and operator training led to better designed human-machine interfaces, better procedures, and a better trained crew. The use of logged data playback allowed almost all vision processing development to take place prior to its initial flight test.

References

[1] R. C. Michelson, "Official Competition Rules for the 1996 Aerial Robotics Competition," Association for Unmanned Vehicle Systems International, 1996.

[2] A. Gelb, "Applied Optimal Estimation," The MIT Press, 1984.

[3] R. K. Heffley and M. A. Minch, "Minimum-Complexity Helicopter Simulation Math Model," NASA CR-177476, 1988.

[4] W. Z. Stepniewsky and C. N. Keys, "Rotary-Wing Aerodynamics," Dover Publications, Inc., 1984.

EFFECTS OF UNSTEADY FAN RESPONSE ON HEAVE DYNAMICS OF AN AIR CUSHION VEHICLE BAG AND FINGER SKIRT

J. Chung* and P. A. Sullivan**
University of Toronto, Institute for Aerospace Studies
Toronto, Ontario, M5S 1A1 , Canada

Air cushion vehicle lift fans are subject to fluctuating delivery pressures generated by vehicle passage over waves and other surface irregularities. This can cause the flow delivered by the fans to deviate significantly from that determined in a steady state fan test. Linear analysis techniques are used to investigate the effects of such unsteady or dynamic fan response on cushion and vehicle dynamics for a simplified cushion configuration representative of the skirt used on large amphibious vehicles. This configuration is a two-dimensional section of a bag-and-finger skirt subject to pure heave surface disturbances. A scaling technique is used to adapt unsteady response data obtained from experiments on a small centrifugal fan, and thus to derive frequency response functions for two configurations. The first is representative of a small laboratory model and the second of a 37 tonne vehicle. The results show significant effect on frequency response, and the dynamic stability of the skirt can be adversely affected.

Introduction

An Air Cushion Vehicle (ACV) is an amphibious vehicle supported by a cushion of pressurized air which is supplied by a lift fan and contained by a flexible skirt. A skirt is a collapsible structure which is inflated and stabilized by the pressure of the cushion air; it enables the vehicle to traverse waves and other surface irregularities while keeping lift air requirements to a minimum. Mainly as a result of the successful development of skirts, aerodynamically propelled ACV's are gradually gaining acceptance in certain special purpose marine roles. For example the Canadian Coast Guard (CCG) uses them for search and rescue, and for navigational aids servicing; the U.S. Navy has a fleet of 91 150 tonne amphibious assault landing craft (LCAC). But such vehicles are not free of problems; one is a tendency to produce a rough ride, and another is a dynamic instability in the skirt most widely used on such vehicles, the *bag and finger* skirt. The work described here is an investigation of the role that dynamic effects in the lift fans plays in cushion dynamics and in contributing to these problems.

Lift fans play a central role in ACV performance and dynamics, and selection of a suitable configuration is a major part of ACV design. Considering first performance, centrifugal fans are the most commonly used, and special designs have been developed to provide the combinations of flow Q_f and delivery pressure p_f required for efficient craft operation.[1] Early investigations of craft and cushion dynamics typically assumed that the lift fan response to p_f fluctuating in time t as a result of motion over surface irregularities is *quasisteady*; that is, the functional dependence of $Q_f(t)$ on $p_f(t)$ is the same as that determined in a steady state *fan characteristic* test. However, investigators found that, even at the relatively low frequencies characteristic of vehicle-dominated motion, lift fans can exhibit true unsteady response. For example, in a 1978 survey of British ACV developments, Wheeler described unsteady or *dynamic* tests on models of ACV fans which the characteristic displayed loops or *hysteresis* characteristic of true unsteady flow.[2] Wheeler noted that these occurred at frequencies which, when scaled to full size, are typical of those encountered when travelling over waves.

Heave, pitch and roll motion of ACV's induced by motion over waves can induce large fluctuations in both p_f and Q_f ; this can extend to momentary flow reversal. Phenomena such as stall and surge can be induced, and this can make fan response very complicated.[3] A feature of the centrifugal

* Research Assistant, Institute for Aerospace Studies, Student member of AIAA.
** Professor, Institute for Aerospace Studies.

Copyright © 1997 by the American Institute of Aeronautics and Astronautics, Inc. All rights reserved.

installations used in the LCAC's is a blade geometry which minimises these effects.[1] However, the typical equilibrium operating condition is usually chosen so that small or moderate fluctuations do not induce such phenomena, in which case one might expect air inertia in the fan inlet, rotor and any delivery ducting to be the main source of unsteady effects. The work presented here uses linear analysis to investigate the effects of unsteady fan response on the heave dynamics and stability of a simplified configuration representative of the bag and finger skirt. The fan dynamic model is scaled from tests on a small centrifugal fan undertaken at the authors' laboratory.[4]

One of the first detailed investigations of unsteady inertia effects in fans was undertaken in 1968 by Ohashi[5]; he used linear cascade theory to investigate an idealized rotor-stator system and obtained a frequency response function $H_{Qp}(f)$ for the effect of oscillations in Q_f at frequency f on p_f. This function had the form $H_{Qp}(f) = H_{Qp}(0)G(f\tau_{RS})$, where $H_{Qp}(0)$ is the slope of the steady state characteristic at the equilibrium operating point, $H_{Qp}(0) = dp_f/dQ_f|_e$, and where τ_{RS} is a time constant. For static stability, $H_{Qp}(0) < 0$. The constant $\tau_{RS} = \cos\lambda_R/(f_B\phi)$, where λ_R is the cascade stagger angle, f_B is a blade-passing frequency and ϕ is a fan flow coefficient. For a centrifugal impeller of diameter D and exit width B rotating at speed N, he used $\phi = Q_f/(\pi NDB)$. The function $G(f\tau_{RS})$ has a complicated structure, but its amplitude response is similar to that of a first order system. However the high frequency phase shift from static behavior approaches 45 degrees rather than 180 degrees. Consequently, in their analysis of a fan-duct system supplying a cushion, Goldschmied and Wormley[6] adapted Ohashi's model by using a function for Q_f in response to p_f in the form of the square-root of a first order lead; that is, with $j = \sqrt{(-1)}$, $H_{pQ}(f) = H_{pQ}(0)(1 + j\pi f\tau_{RS})^{1/2}$. Here $H_{pQ}(0) = [H_{Qp}(0)]^{-1}$. Ohashi undertook experiments using a centrifugal pump and water as the working fluid; however the results followed his predictions only qualitatively, suggesting that other factors are involved.

Ohashi did not consider unsteady inertia effects in the outlet volute or other ducting associated with the pump he tested. In 1978 Durkin and Leuhr[7] obtained a limited amount of response data for a model of an ACV fan installation. They suggested that their data could be represented by a first order lag; $H_{pQ}(f) = H_{pQ}(0)/(1 + jf\tau_F)$, where τ_F is chosen to fit the data. Such a response is produced by unsteady inertia in a duct; if the duct has length L_d and cross-sectional area A_d, $\tau_F = -2\pi H_{pQ}(0)I_d$, where $I_d = \rho L_d/A_d$ is the inertance of the duct air. Durkin and Leuhr used the fan geometry to estimate an equivalent I_d; this gave computed values of τ_F sufficiently close to measured values to imply that equivalent duct inertance effects are a major, even principal source of unsteady fan effects. However, their experiments suggest that the effective inertance depends on fan speed.

Sullivan, Hinchey and Green show that even moderately short ducts can have a major effect on dynamics and stability so that, if these are present in an ACV fan installation, a lumped inertance model should give an accurate representation of air supply system dynamics at the low and moderate frequencies characteristic of vehicle-dominated motion.[8] But, for fan installations in vehicles of the type used by the CCG and the U.S. Navy a centrifugal rotor either delivers the air directly into a plenum chamber or uses only on outlet scroll. For such, configurations the previous work suggests that it is difficult to accurately predict the response of a given geometry. Sullivan, Gosselin and Hinchey proposed a technique for experimentally determining frequency response of a model of an ACV fan installation, and used it to obtain data for a small centrifugal fan.[4] They showed that, for a small cushion model, unsteady effects had a major effect on frequency response.

The work described here uses scaling arguments to adapt Sullivan Gosselin and Hinchey's data to the prediction of frequency response and stability of a simplified configuration representative of the bag and finger skirt. The static behavior of this configuration has been investigated theoretically and experimentally by Sullivan, Charest and Ma[9]; the nonlinear and linear dynamics with quasisteady fan response has been investigated respectively by Chung, Sullivan and Ma,[10] and by Chung and Sullivan.[11] The latter linear dynamic analysis is extended to include unsteady fan effects, and frequency response and stability results are presented for two cases: the first is representative of the laboratory model used by Sullivan and Charest[9] in their experiments, and the second is representative of a 37 tonne vehicle used by the CCG.

Unsteady Fan Experiment and Results

Direct determination of the true unsteady flow delivered by a fan is a difficult task; consequently Sullivan, Gosselin and Hinchey[4] proposed an indirect method based on the air cushion experiment depicted in the Fig. 1. The fan being tested (1), together with its inlet and any downstream supply ducting delivers air to a rigid-wall plenum chamber cushion (2) which, in turn is mounted over a servohydraulically driven heave table (3). With the table oscillating at a preset frequency and

amplitude, the histories of the pressure $p_c(t) = p_f$ and the hovergap $h(t)$ between the lip of the cushion and the table are measured and recorded.

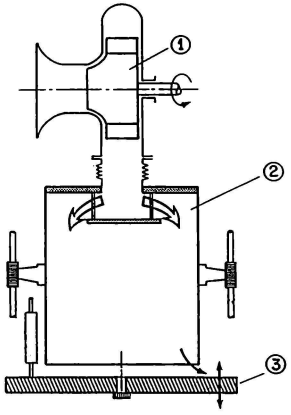


Fig. 1 Modified air cushion heave dynamics experiment used to obtain fan response data.

The volume flux history $Q_f(t)$ is determined as follows. First the volume flux $Q_a(t)$ of air escaping from the cushion to atmosphere is assumed quasisteady, so that it can be calculated by the usual orifice law:

$$Q_a(t) = C_D L_p h_e(t) \sqrt{\frac{2p_c(t)}{\rho_a}} \qquad (1)$$

In this expression C_D is a discharge coefficient, L_p is the peripheral length of the cushion lip and ρ_a is atmospheric air density. For the values of h_e and f used in these experiments, the quasisteady approximation is believed to be very accurate.[4,8] Then, to determine $Q_f(t)$, one applies the equation of continuity to the plenum chamber volume:

$$Q_f(t) = Q_a(t) + C_c \frac{dp_c}{dt} + S_c \frac{dh}{dt} \qquad (2)$$

In this expression C_c is the pneumatic capacitance of the plenum chamber volume, and S_c is the footprint area of the cushion. The capacitance $C_c = V_c/\gamma P_a$, where V_c is the plenum chamber volume, γ is the ratio of specific heats for air, and P_a is the absolute atmospheric pressure.

Tests were conducted on a small centrifugal fan having $D = 0.330$ m., and $B = 0.120$ m. for fan speeds in the range $2000 \le N \le 3000$ rpm.[4] The table frequency was in the range $0 \le f \le 12$ Hz, corresponding to reduced frequencies $\bar{f} = f/N \le 0.36$. Figure 2 compares the static characteristic for this fan with the mean response obtained during the dynamic tests; it is in the form of a plot of $C_{FP} = p_f/(\rho_a N^2 D^2)$ against $C_{FQ} = Q_f/(ND^3)$. The curve is a polynomial fit to steady state data obtained by removing the plenum chamber and installing an orifice-plate flow meter, and the data points are the mean values of the measured p_f and computed Q_f histories averaged over the values of f used in the tests. With the fan response expressed in the form $H_{PQ}(f) = M_{PQ}(f)\exp[j\phi_{PQ}(f)]$, tests at selected frequencies showed that both gain M_{PQ} and phase shift ϕ_{PQ} are independent of the amplitude of oscillations in $h_e(t)$ for amplitudes up to 50 percent of the mean h_e. Both this amplitude independence and the agreement of the dynamic means with the static characteristic confirm that the fan response to the oscillations in p_f is linear.

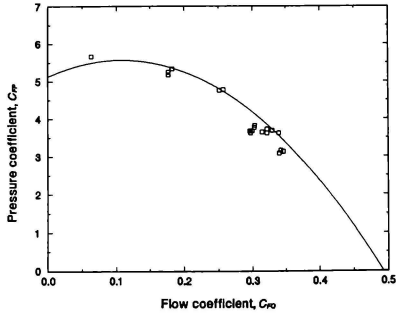


Fig. 2 Comparison of dynamic test mean values of p_f and Q_f with static characteristic.

Dimensional arguments suggest that, for a family of geometrically similar fans, if the dynamic response is linear, and if viscous effects can be ignored, then with $\overline{M}_{PQ} = [(\rho_a N)/D]M_{PQ}$ being a nondimensional gain,

$$\overline{M}_{PQ},\ \phi_{PQ} = fns(\bar{f}, C_{FQ}) \qquad (3)$$

To test this hypothesis Sullivan, Gosselin and Hinchey plotted $\overline{M}_{PQ}$ and ϕ_{PQ} against $\bar{f}$ for $N = 2000$ and 3000

rpm and for nominally the same C_{FQ}. Figures 3 and 4 gives this data; the close agreement supports the hypothesis. It is the data in Figs. 3 and 4 which is used to construct frequency response functions used in the present analysis.

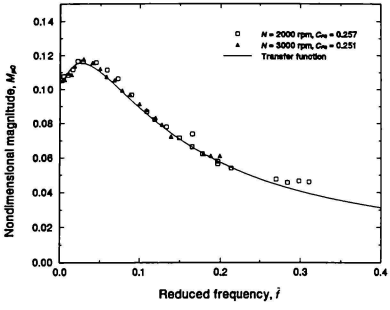


Fig. 3 Nondimensional magnitude response of test fan plotted against reduced frequency.

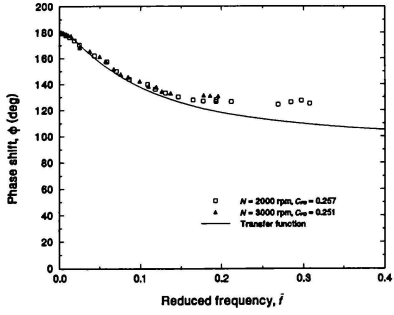


Fig. 4 Phase response of test fan as a function of reduced frequency.

Simplified Bag-Finger Skirt Model

The geometry of the bag-and-finger skirt is complex and can undergo large changes during vehicle motion. Consequently, to allow formulation of the dynamics from first principles, Ma and Sullivan[12] proposed investigation of the configuration depicted in Fig. 5. It is a two-dimensional section of the skirt which is subject to pure heave disturbances induced by vertical motion of a flat horizontal surface moving according to a specified law $h_g(t)$. The two main skirt elements are the *bag ABCE* and the *fingers CDF*; both are made from an elastomer-coated fabric. In the full-scale craft the bag extends around the periphery of the vehicle base; in the model it is replaced by two cylindrical loops. The fingers are folded from an approximately triangular piece of material such that a horizontal section through them is U-shaped. They are attached to the bag at *C*, and to the vehicle base at *E*. Also the interior compartmentation required for pitch and roll stability of a craft is omitted.

The lift fan delivers air directly into a plenum chamber of which the bags form a part, consequently the *bag pressure* $p_b = p_f$. The difference between p_b and the pressure p_c in the cushion causes air to flow from the bag to the cushion through *bag orifices* located in the inner part of the bag *CE*; there is usually one orifice for each finger. Typical operating conditions are such that $p_b = 1.2p_c$ approximately. In the full-scale craft the bag filters out long-wave disturbances and the fingers filter the short-wave disturbances; the shape of the latter also ensures that they maintain the cushion seal by expanding when one is torn or damaged.

For the configuration in Fig. 5, the function of the skirt may be described as follows. Changes in h_g modulate h_e, or the distance between the bottom tips of the fingers *F* and the ground-plane. This, in turn, modulates Q_e and hence p_c and the flow Q_b from bag to cushion. The change in p_c alters the ratio p_b/p_c and thus the bag shape and *skirt height* h_r. Changes in h_r minimise fluctuations in h_e and thus p_c, thus minimising fluctuations in the force acting on the vehicle base. This force is also modulated by changes in the skirt *lateral displacement* x_r; although this effect is small for full sale configurations, it can be large for laboratory models such as that investigated by Sullivan, Charest and Ma.[9]

Governing Equations - Vehicle and Cushion

Figure 6 depicts the main elements of the skirt model used in the present analysis and defines the geometric parameters required.[9,10] The model consists of three distinct parts. The first is the *outer bag*, which is the part *ABC* of the skirt between the upper attachment point *A* and the outer edge of the fingers, *C*. The second part is the *inner bag BDE*, and the third is the fingers *CDF*. The outer bag is assumed to be a

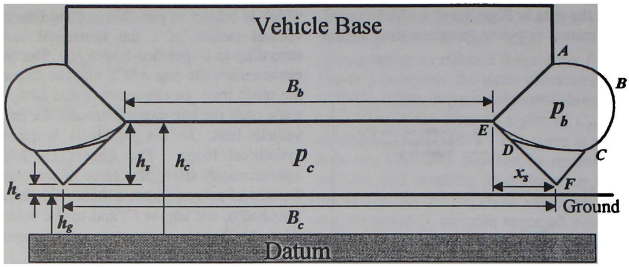


Fig. 5 Simplified bag-and-finger skirt model used in the present analysis.

massless inelastic membrane subject to p_b, so that it is always a circular arc. The inner bag is assumed to comprise two rigid massless links having lengths L_1 and L_2 respectively; these links are subject to the pressure difference p_b - p_c. The fingers are modelled as rigid bodies attached to the inner bag at C and D; the part CF is subject to p_c. The skirt has its mass concentrated at the centroid of the finger, and has two degree-of -freedom corresponding to the angles α and γ defined in Fig. 6.

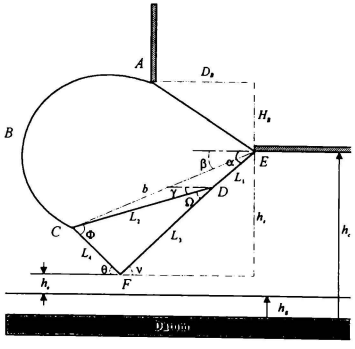


Fig. 6 Elements of skirt model used in present analysis.

The basic motion variables required to describe the heave dynamics are depicted in Fig. 5. With the vehicle base height being $h_c(t)$,

$$h_g + h_e + h_s = h_c \tag{4}$$

where, from Fig. 5, h_s is given by

$$h_s = L_1 \sin\alpha + L_2 \sin(\Omega + \gamma) \tag{5}$$

With L_m being the length of the model normal to the diagram in Fig. 5,

$$S_c = L_m B_c = L_m(2x_s + B_b) \tag{6}$$

where B_b is the width of the vehicle base between bag attachment points.

The analysis also allows for the possibility that the skirt contacts the ground, corresponding to $h_s < 0$. Consistent with observations of such systems, when this occurs, it is assumed that skirt collapse occurs only locally, with the finger geometry being that above the intersection of the uncollapsed finger and the ground -plane.

The equations of motion of the skirt and craft are formulated using a Lagrangian technique[12]. That governing the craft motion is

$$M_c\ddot{h}_c + M_s[L_1(\sin\alpha\ddot{\alpha} - \cos\alpha\dot{\alpha}^2) + L_M(\sin\gamma_M\ddot{\gamma} - \cos\gamma_M\dot{\gamma}^2)] + M_c g = p_c S_c \tag{7}$$

and those governing the skirt motion are:

$$M_s L_1[L_1\ddot{\alpha} - L_M\sin(\gamma_M - \alpha)\dot{\gamma}^2 + L_M\cos(\gamma_M - \alpha)\ddot{\gamma} - \ddot{h}_c\cos\alpha - g\cos\alpha] = p_b V_\alpha^{ob} + (p_b - p_c)V_\alpha^{ib} + p_c V_\alpha^{f} \tag{8}$$

$$M_s L_M[L_M\ddot{\gamma} + L_1\sin(\gamma_M - \alpha)\dot{\alpha}^2 + L_1\cos(\gamma_M - \alpha)\ddot{\alpha} - \ddot{h}_c\cos\gamma_M - g\cos\gamma_M] + I_s\ddot{\gamma} = p_b V_\gamma^{p_b} + (p_b - p_c)V_\gamma^{p_b} + p_c V_\gamma^{f} \quad (9)$$

In these equations, M_c and M_s are, respectively, the total craft and skirt masses, g is the acceleration due to gravity, and the other terms on the left hand sides are defined in Fig. 6. The terms on the right-hand sides of Eqs. 9 and 10 are, in order, the effect of p_b acting on the outer bag, $(p_b - p_c)$ acting on the inner bag, and p_c acting on the part CF of the finger in Fig. 6. The six quantities $V_\alpha^{p_b} \ldots V_\gamma^{f}$ are functions of α and γ, they are derived by using the result from hydrostatics that, if a surface sweeps out a volume δV under the action of a pressure p, then the work done by p is $p\delta V$. They are listed in Refs. 9 and 10.

The fluid mechanics of the skirt system is described by laws governing the flows Q_b and Q_a, and conservation laws equivalent to Eq. 2 for the bag and cushion volumes, $V_b(\alpha,\gamma)$ and $V_c(\alpha,\gamma,h_c)$ respectively. The flows Q_b and Q_a are assumed to obey the quasisteady orifice flow laws[13]

$$Q_c = A_b\,\mathrm{sgn}(p_b - p_c)\sqrt{\frac{2|p_b - p_c|}{\rho}} \quad (10)$$

$$Q_a = L_p h_f(h_e,\theta)\,\mathrm{sgn}(p_c)\sqrt{\frac{2|p_c|}{\rho}} \quad (11)$$

In these expressions A_b is the total effective bag orifice area, including any discharge coefficient, and h_f is an effective or equivalent slot width, which is a function of both h_e and the external slope of the finger relative to the ground θ shown in Fig. 6. The latter quantity is expressed in the form

$$h_f = B_f f_h(\frac{h_e}{B_f},\theta) \quad (12)$$

where B_f is the finger width. The function f_h is the product of the area between the bottom of the fingers and the surface with a discharge coefficient dependent on θ. For $h_e > 0$, f_h depends linearly on h_e. When surface contact occurs at $h_e = 0$, further decrease in h_e causes h_f to decrease nonlinearly as the circular-arc tips of the fingers (at F in Fig. 6) collapse, shutting off the flow. Equations 10 and 11 allow for flow reversal.

The bag and cushion conservation laws are:

$$C_b\dot{p}_b + \dot{V}_b = Q_b - Q_c \quad (13)$$

$$C_c\dot{p}_c + \dot{V}_c = Q_c - Q_a \quad (14)$$

In these equations, C_b and C_c are respectively, the pneumatic capacitances of V_b and V_c, given respectively by $C_b = V_b/(\gamma P_a)$ and $C_c = V_c(\gamma P_a)$; the expressions for V_b and V_c in terms of α,γ and h_c are given in Ref. 10.

Equations 4-14 are linearized and transformed to the frequency domain using standard techniques.

Lift Fan Model

The analytical model is completed by specifying the functional dependence of $Q_f = Q_b$ on p_b. In the analysis of Ref. 10, this takes the form

$$p_b = f(Q_b) = p_{br}\{\xi - (\xi - 1)(\frac{Q_b}{Q_{br}})^3\} \quad (15)$$

In this expression p_{br} and Q_{br} are, respectively, a reference pressure and a reference flow. They are usually taken to be those at the craft's operating equilibrium conditions. The parameter ξ is the ratio p_b/p_{br} corresponding to $Q_b = 0$; usually $1.2 \le \xi \le 1.4$ to ensure adequate static stability margin. The form of equation (15) reflects the fact that flow reversal can occur; it is chosen so that $dp_b/dQ_b \le 0$ over the expected operating range, including when $Q_b < 0$. This equation provides the quasistatic fan condition which is a reference in the present calculations. In this case $H_{pQ}(f)$ is taken to be $H_{pQ}(0) = [dp_f/dQ_f]_e^{-1}$

Development of a dynamic fan response model for the laboratory and 37 tonne configurations proceeded in two stages. First, an analytical fit to the data in Figs. 3 and 4 was obtained in a form suitable for scaling. The gain data in Fig. 3 was normalised by its static value, $M_{pQ}(0)$; this normalised gain data, together with the phase data in Fig. 4 was then manually fitted with a transfer function of the form

$$H_{pQ}(\bar{s}) = -M_{pQ}(0)G_f(\bar{s}) = -M_{pQ}(0)\left[\frac{1 + A_1\bar{s}}{1 + B_1\bar{s} + B_2\bar{s}^2}\right] \quad (16)$$

In Eq. 16, with $\omega = 2\pi f$, and $s = \sigma + j\omega$ being the Laplace variable, $\bar{s} = s/N$. For the above data, $A_1 = 76.9$, $B_1 = 75.0$, and $B_2 = 625$; the fit is plotted in Figs. 3 and 4.

The second stage proceeded in four steps. The first step was to select a total cushion equilibrium flow rate Q_e appropriate to the two configurations. In the case of the laboratory model Q_e was that actually used

in the experiments and, for the 37 tonne configuration, Q_e was specified using a currently accepted design rule.[10] The second step was to choose values of D and N, together with a number N_R of fan rotors, to deliver the specified Q_e while keeping the value of C_{FQ} per rotor equal to the average of that used in the tests of Figs. 3 and 4, namely $C_{FQ} = 0.255$. The third step was to use Eq. 15 to determine $H_{pQ}(0)$ for the specified Q_e. Finally, the $H_{pQ}(f)$ for the two configurations is the product of this $H_{pQ}(0)$ and $G_f(\bar{f})$, where $\bar{f}$ is now based on the value of N selected for the fans used in the configurations.

Table 1 gives the parameters for the two configurations; the 37 tonne configuration was adapted from the CCG vehicle *Waban-Aki*, which has four double rotor lift fans with D = 0.885 m. and which typically operates at N = 2300 rpm.

Configuration	Lab. Model	37 tonne configuration
Number of rotors, N_R	1	14
Rotor Diameter, D (m)	0.33	0.885
Fan speed, N (rpm)	2,500	2,300
Equilibrium flow, Q_{be}(m^3/s)	0.57	96.8
Equilibrium bag pressure, p_{be}(Kpa)	0.661	2.419
Static gain, $M_{pQ}(0)$	3.089E-4	0.032

Table 1 Fan parameters used in laboratory model and 37 tonne configurations.

Results

For the two configurations Table 2 gives base and skirt parameters together with the equilibrium hover conditions. Figures 7 - 10 compare the effect of quasisteady and unsteady fan dynamics on the amplitude and phase response for these configurations. The amplitude response curves in Figs. 7 and 9 display the characteristic double resonance for this skirt system; the low frequency peak is associated with the craft mass, and the high frequency peak is associated with the skirt mass.[10] Indeed, the response in Fig. 9 displays a characteristic problem for the bag-and-finger skirt: it can produce large craft heave response in a frequency range in which humans are particularly sensitive, namely 4 - 8 Hz.

Configuration	Lab. Model	37-tonne configuratiom
L_m (m)	0.914	21
B_b (m)	0.57	5.75
D_b (m)	0.18	1.39
H_b (m)	0.15	0.98
L_{ab} (m)	0.37	2.6
L_1 (m)	0.05	0.18
L_2 (m)	0.36	2.02
L_3 (m)	0.3	1.69
L_4 (m)	0.2	1.1
B_f (m)	0.09	0.38
M_s (kg)	1.11	900
M_c (kg)	44	36,740
h_c (m)	0.17	1.26
h_s (mm)	11	4
p_c (kPa)	0.395	2.02

Table 2 Configurations and equilibrium conditions used in simulations.

For both configurations unsteady fan effects on the magnitude response are negligible in the neighborhood of the craft mass resonance, but have a major impact on the skirt mass resonance. However, the effects on the two configurations are markedly different. For the laboratory model, the magnitude of the peak is increased by a factor of 3.3 and the natural frequency is increased slightly; in contrast, for the 37 tonne configuration, the peak is decreased by a factor of 7.7, and the natural frequency increases slightly. In relation to the ride comfort problem, Chung, Sullivan and Ma's quasisteady fan analysis suggests that reducing skirt mass can significantly improve the problem.[10] Figures 11 and 12 shows the effect of reducing M_s from 900 kg to 450 kg for quasisteady and unsteady response; this suggests that unsteady fan dynamics offsets the improvement that might be expected by reducing M_s.

Considering now the phase response, the laboratory model is only slightly affected over the frequency range plotted, but the 37 tonne configuration shows a major shift in the vicinity of the skirt resonance, suggesting that unsteady fan dynamics has a destabilizing effect. Now the characteristic limit cycle

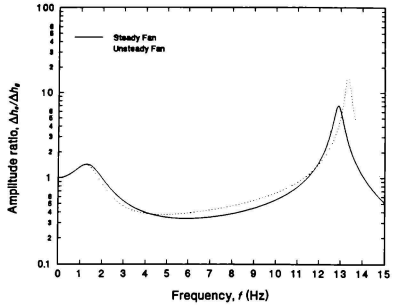


Fig. 7 Unsteady fan effects on the heave amplitude response for laboratory model

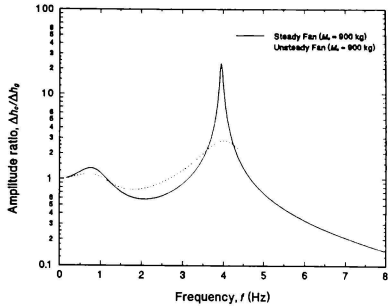


Fig. 9 Unsteady fan effects on the heave amplitude response for full scale 37 tonne craft.

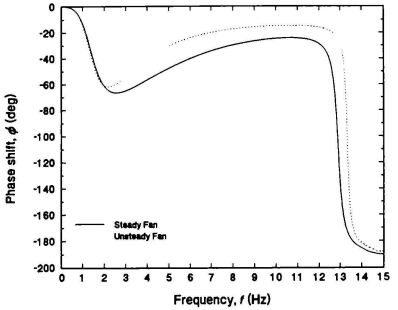


Fig. 8 Unsteady fan effects on the heave phase response for laboratory model

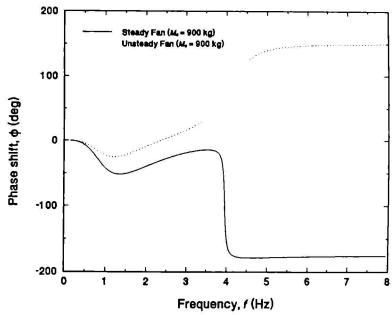


Fig. 10 Unsteady fan effects on the heave phase response for full scale 37 tonne craft.

oscillation of the bag-and-finger skirt known as *skirt bounce*[1] is basically a vertical motion of the bag; the analysis of Ref. 13 shows that this is associated with a dynamic instablity in the *skirt-cushion* system; that is in the dynamics that occurs when h_c is fixed. Figure 13 plots the eigenvalues for the 37 tonne configuration and for quasisteady and unsteady fan dynamics; examination of the eigenvectors corresponding to these eigenvalues shows that the pair S_1 and U_1 are associated mainly with large variation in the skirt angle α in Fig. 6 which governs vertical skirt motion. This implies that unstedy fan dynamics contributes to the onset of skirt bounce in this configuration.

Discussion and Conclusions

The results presented above imply that unsteady fan dynamics has a major effect on the dynamics of the bag-and-finger skirt; this, in turn, can have a significant impact on the dynamics of any vehicle using that skirt. But they also suggest that no simple rules can be discerned; the effects appear to depend on the size and other attributes of any given vehicle. Thus, for the 37 tonne configuration investigated here, the results suggest that a ride-comfort problem might be considerably alleviated; however, it would be premature to draw equivalent conclusions for

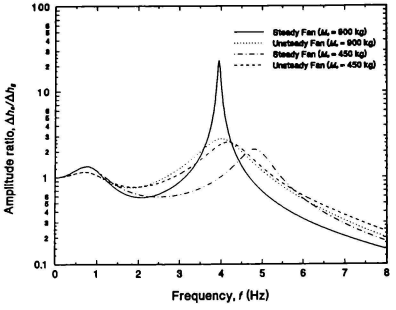


Fig. 11 Effect of reducing M_s for quasisteady and unsteady response on the heave amplitude response for full scale 37 tonne craft.

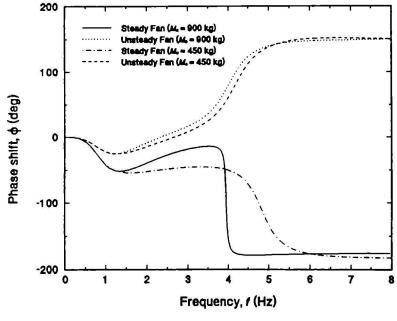


Fig. 12 Effect of reducing M_s for quasisteady and unsteady response on the heave amplitude response for full scale 37 tonne craft.

vehicles the size of the U.S. Navy's LCAC.

In this regard the results presented for the laboratory model should be accurate because they refer to the fan actually used in testing that model. However, the rotor blade geometry differs in significant details from that used on the CCG vehicles, and on the LCAC's. Consequently the present results can only be regarded as a guide. But they underscore the need to include a realistic dynamic fan model in any analysis of ACV dynamics.

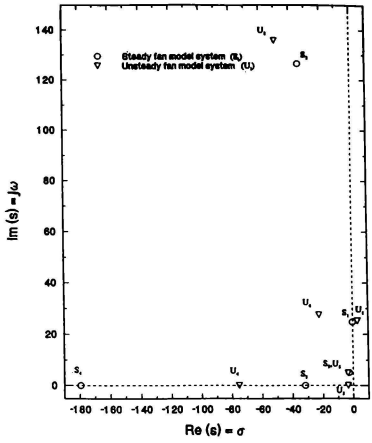


Fig. 13 Eigenvalues for the system with quasisteady and unsteady response for full scale 37 tonne craft.

Acknowledgement

The present research was supported in part by the Natural Sciences and Engineering Research Council of Canada.

References

[1]Mantle, P. J., "Design Fundamentals," *Hovercraft Technology, Economics and Applications,* J. R. Amyot, Editor, Elsevier Studies in Mechanical Engineering, Vol. 11, Elsevier, Amsterdam, 1989, pp 73-134.

[2]Wheeler, R. L., "Recent United Kindom Hovercraft Development," *Journal of Hydronautics*, Vol. 12, No. 1, Jan. 1978, pp. 3-17.

[3] Moran, D. D. and Bullock, M. V., "Dynamic Response of Lift Fans Employed in Hovering Craft: A Review of the Literature," *Hovering Craft and Hydrofoil*, Vol. 16, No. 7/8, April-May 1977, pp. 13-22.

[4]Sullivan, P. A., Gosselin, F., and Hinchey, M. J., "Dynamic Response of an Air Cushion Lift Fan," *Proceedings of the 1992 Intersociety High Performance*

Marine Vehicles Conference (American Society of Naval Engineers), June 1992, pp. ACV39-47.

[5]Ohashi, H., "Analytical and Experimental Study of the Dynamic Characteristics of Turbopumps," NASA Technical Note TN D-4298, April 1968.

[6]Goldschimied, F. R. and Wormley, D. N.,"Frequency Response of Blower/Duct/Plenum Fluid Systems," *Journal of Hydronautics*, Vol. 11, No. 1, Jan.1977, pp. 18-27.

[7]Durkin, J. M. and Luehr, L. H.,"Dynamic Response of Lift Fans Subject to Varying Back Pressure," *American Institute of Aeronautics and Astronautics/Society of Naval Architects and Marine Engineers Advanced Marine Vehicle Conference*, San Diego, CA, April 17-19, 1978, AIAA Paper 78-756.

[8]Sullivan, P. A., Hinchey, M. J., and Green, G. M., "A Review and Assessment of Methods for Prediction of Dynamic Stability of Air Cushions," *Journal of Sound and Vibration*, Vol. 84, No. 3., Oct. 1982, pp. 337-358.

[9]Sullivan, P.A., Charest, P. A., and Ma, T., "Heave Stiffness of an Air Cushion Vehicle Bag and Finger Skirt," *Journal of Ship Research*, Vol. 38, No. 4, Dec. 1994, pp. 302-307.

[10]Chung, J., Sullivan, P. A., and Ma, T., "Nonlinear Heave Dynamics of an Air Cushion Vehicle Bag and Finger Skirt," *AIAA Flight Simulation Technologies Conference*, San Diego, CA, July 29-31, 1996, AIAA paper No. AIAA - 96 - 3482.

[11]Chung, J., and Sullivan, P. A., "Linear Heave Dynamics of an Air Cushion Vehicle Bag and Finger Skirt," Japan Society for Aeronautical and Space Sciences 10th International Sessions in 34th Aircraft Symposium, Tottori, Japan, Oct. 1996.

[12]Ma, T., and Sullivan, P. A., "Dynamics of Responsive Skirts," *Hovercraft Technology, Economics and Applications*, J. R. Amyot, Editor, Elsevier Studies in Mechanical Engineering, Vol. 11, Elsevier, Amsterdam, 1989, pp 526-600.

[13]Sullivan, P. A., Byrne, J. E., and Hinchey, M. J., "Nonlinear Oscillations of a Simple Flexible Skirt Air Cushion," *Journal of Sound and Vibration*, Vol. 102, Oct. 1985, pp. 269-283.

SIMULATION OF FIXED-WING AIR-TO-AIR COMBAT USING ITEMS

Dr. P.B. Howells, Dr. M. Charbonneau and Mr. B.Kwan
CAE Electronics Limited
St-Laurent, Quebec, Canada

Abstract

This paper describes the use of CAE's Interactive Tactical Environment Management System (ITEMS) for use in Air-to-Air combat pilot training. Details are given of the principal software components which provide for off-line scenario creation and on-line real-time execution. The maneuvers supported by ITEMS provide for training in Guns, IR-missile and Radar-missile engagements. The instructor can create a sequence that comprise of all three engagement modes if required. The system supports two full six-degree of freedom aerodynamic models that provide for training in two-vs-one and one-vs-two engagements. A wingman capability also exits where the leader maybe the manned simulator or a second computer generated target.

Introduction

With ever increasing cuts in defence spending, maintaining a high degree of defence readiness in air-to-air combat operations will be difficult. One alternative is to supplement air combat training with simulator time. Effective training, however, requires that the air targets have representative characteristics and performance in terms of maneuvering, aircraft flight dynamics and decision process.

To support training in air-to-air combat, CAE has designed an Interactive Air Target (IAT) as part of its Interactive Tactical Environment Management System (ITEMS). The system provides training in the areas of weapons deployment, basic fighter maneuvering, two-vs-one and one-vs-two engagements. For weapons training, the air target can be programmed to perform flight paths composed of simple weaves and turns, where the student is required to track the target and obtain a firing solution using guns or missiles. Control over the "g" capability of the target enables the instructor to make the task more or less difficult for the student. The basic maneuvers that form part of the ITEMS system may be used for training in basic fighter maneuvering. The air target, for example, may be set to perform a pursuit maneuver which the manned simulator has to counter.

The air target is controlled by a programmable rule base. The rules are defined using condition and response parameters selected from a pull-down menu. For example, the condition parameters for the IAT to perform a guns engagement are for the opponent to be in the front hemisphere and in guns range. The response parameter in this case would be to fire the gun. The air target responds continuously to the rules that are triggered as the air target switches from offensive to defensive positions depending on the engagement geometry.

The air target considers the full equations of motion in the longitudinal axis with some simplifications in the lateral axis. The current system supports two such targets, although more could be added if required. The targets are defined using an off-line Database Management System (DBMS) for the entry of the principal aerodynamic data (lift and drag) and characteristics of the propulsion system. A range of weapon systems and sensors can be assigned to the aircraft. Typical aircraft types supported by ITEMS include F4-Phantom, Tornado, MiG-29 Fulcrum, and F15-Eagle.

This paper gives an overview of the ITEMS software package and how the air combat simulation is implemented.

ITEMS Components

The objective of this paper is to provide details of how ITEMS can be used for pilot training in air to air combat. However, it is necessary first to describe ITEMS in global terms since ITEMS is also used to equip the air target with a range of sensors, weapons and countermeasures that will be used in the simulation. This overview is given below. Key to the description is the functional elements and how they interact.

Copyright © 1997 by the American Institute of Aeronautics and Astronautics, Inc. All Rights Reserved.

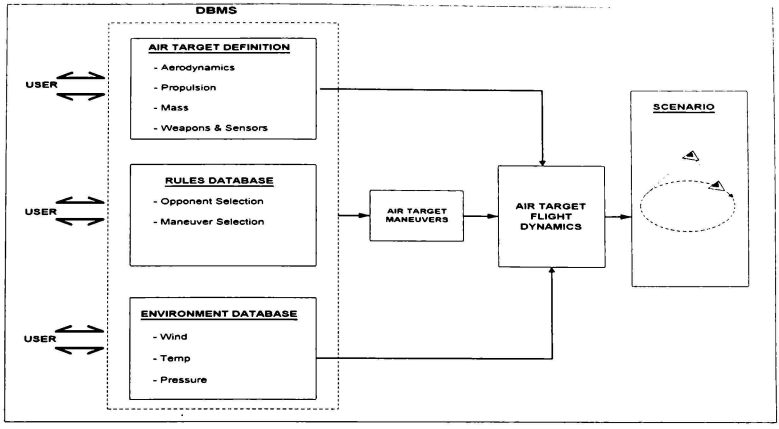


Figure 1 ITEMS Software Components

a) ITEMS Overview

ITEMS provides a user-definable environment without any hard-coded effects. High fidelity models for the threat and friendly forces are used throughout, and can be adapted to different types of tactical environments; air-to-air and air-to-ground missions for example. The software is fully modular and supports Distributed Interactive Simulation (DIS) network interconnectivity.

The method of defining a tactical scenario is a bottom-up approach, where the elementary element databases (weapons, sensors, and countermeasures) of the scenario are first defined. These elements are then combined to generate higher level databases that allow for a complete tactical environment by combining all elements and pertinent data. Access to the library of databases is provided via dedicated X-window based graphical user interfaces. Version 4.0 of ITEMS provides for the creation of Air, Land and Sea scenarios by supporting such elements as fixed-wing aircraft, helicopters, ground vehicles, missile sites and gun emplacements.

The principal components of the ITEMS system are illustrated in Figure 1. They include the air target definition, air combat maneuvering database, rules database, the weapons and sensor database, and the environment database. The function of each of these components is described below.

b) Air Target Definition

The air target is defined using an off-line utility called the Database Management System, or DBMS. Written in X-Motif, the DBMS editor allows the user to enter the data required to define the air target. Typical parameters include lift and drag coefficients, engine thrust characteristics, mass and wing area.

c) Rules Database

Each player in a scenario is assigned a rule set. Usually there are two: a prime opponent selection rule set and a maneuver rule set. The opponent selection rule set contains the rules for selecting an opponent, i.e. should fast moving jets take priority over slower moving transport aircraft. The maneuver rule set is the decision kernel that invokes the maneuvers from within the air combat maneuvering database mentioned above. It is the rules defined here that instruct the air target to maneuver either offensively, or defensively. The logic to transition from one maneuver to another is also part of this rule base, i.e. when to switch from, pursuit to gun track and finally disengagement.

Copyright © 1997 by the American Institute of Aeronautics and Astronautics, Inc. All Rights Reserved.

d) Air Combat Maneuvering Database

The air combat database comprises a series of maneuver routines that can be invoked by the rule base. The rule base tells the system which maneuver to perform and the air combat database carries out that maneuver. The current system supports upwards of fifteen different maneuvers that include High "G" Barrel Roll, Immelmann, Pure Pursuit, Gun Tracking, Lift Vector Turn, Scissors, Jinks, Break, High Yoyo, and Low Yoyo. These maneuvers cover the complete spectrum of offensive maneuvering, defensive maneuvering and stalemate.

e) Weapons and Sensors Database

The weapons and sensors database is used for specifying the characteristicsof the weapons and sensors. All models within ITEMS are based on physical models that consider the principal factors that affect performance. For example, in the case of the missile the user would be requested to specify the mass and aerodynamic characteristics together with the guidance mode. Countermeasures are defined in a similar manner with specification of radar cross section in the case of chaff and intensity in the case of IR flares. The weapons and sensors are assigned to the air target when the player is created. Typical examples of the kind of systems supported include:

i) sensors
- Radar warning receiver
- Electro-optical sensors
  - visual (pilot eyes)
  - thermal (FLIR,IRST)
  - image intensifier
  - Low light TV
  - TV camera
- Radar
- Laser designator
- Laser range finder

ii) countermeasures
- Heat flare
- Light flare
- Smoke
- Chaff
- IR jammer
- Laser jammer
- Radar jammer
- Radio jammer

iii) weapons
- Guns
- Air defence artillery
- Rockets
- Bombs
- Missiles

f) Line Of Sight System (LOS)

The LOS system checks if a valid line of sight exists between any two players. This is derived from the LOS database which is processed from, and therefore correlated with, the visual database. A polygon representation is used which provides a significant improvement in accuracy over the traditional grid post method. The effect of the earths curvature is also taken into account.

g) Environment

The atmospheric environment is controlled within the scenario by defining wind velocity, temperature and pressure as a function of altitude.

Air Target Modelling

The Interactive Air Target (IAT) is a full six degree of freedom aerodynamic model based on the Euler equations of motion. The limits of performance are derived from physical characteristics such aerodynamic loads and engine thrust. The data are entered through the off-line DBMS system. Typical parameters include:

- the aircraft mass
- the structural positive and negative acceleration limits
- the maximum pitch rate
- the wing geometry
- the body moment of inertia about the Y-axis
- the speed brake drag factor and the ram drag factor
- the basic drag coefficient as a function of incidence angle and Mach number
- the basic lift coefficients vs incidence angle
- the lift coefficient due to elevator incidence angle
- the lift coefficient due to pitch moment

Copyright © 1997 by the American Institute of Aeronautics and Astronautics, Inc. All Rights Reserved.

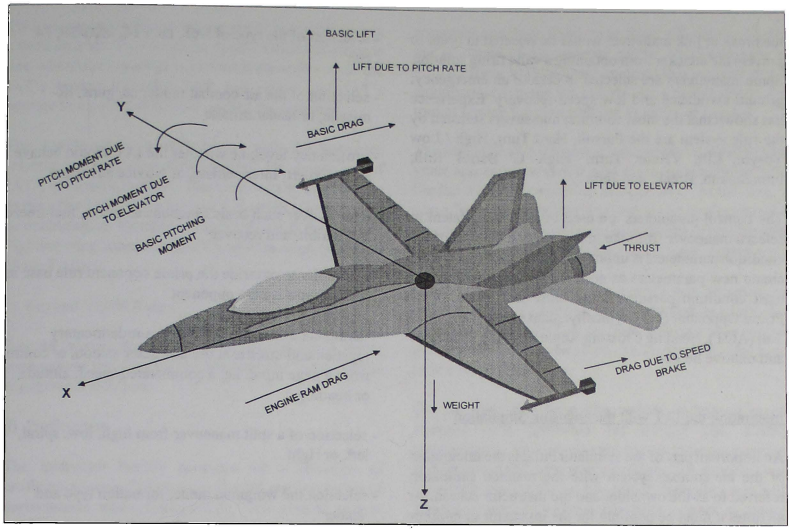


Figure 2 Aircraft Applied Forces and Moments

- the basic pitch moment vs incidence angle
- the basic pitch moment due to elevator angle
- the pitch moment due to pitch rate
- the engine thrust data (the idle, military and afterburners thrust and an altitude correction)

At scenario load time, the IAT parameters are loaded into memory. At run-time, the aerodynamic coefficients and engine thrust rating data are interpolated based on aircraft state.

The motion path and attitude of the IAT is derived from the applied forces and moments illustrated in Figure 2. which appear as parameters in the Eulers equation of motion. As shown in the figure, there are contributions from the effects of gravity, aerodynamic forces and the propulsion system. The axis systems employed include the stability axis, in which all aerodynamic forces are calculated, the body axis and the inertial axis.

The aerodynamic forces are computed from the aerodynamic coefficients entered via DBMS and ambient conditions at the aircraft. Contributions from the control surfaces are also considered. The engine thrust is also computed from the data entered via DBMS and the commanded throttle position.

Air Target Maneuvers

As indicated in Figure 1, the air combat rules system selects a maneuver from the maneuver list. When the selection is made, the maneuver routine proceeds to compute the position of the control surfaces and throttle position. The resulting aerodynamic loads and thrust are used to maneuver the aircraft to the desired trajectory, resulting in such maneuvers as the pursuit, yoyo, and hard turn.

The current library provides for a wide range of offensive, defensive and stalemate maneuvers. An aggressive fighter will probably use the pursuit maneuver at the start of the engagement outside of weapons range. Alternatively, when fending of an attack from the rear

Copyright © 1997 by the American Institute of Aeronautics and Astronautics, Inc. All Rights Reserved.

the break or jink maneuver would be selected in order to prevent the attacker from obtaining a valid firing solution. Some maneuvers are selected in case of an emergency: ground avoidance and low speed recovery. Experience has shown that the most common maneuvers selected by the rule system are the Pursuit, Hard Turn, High / Low Yo-yo, Lift Vector Turn, High G Barrel Roll, Immelmann, Break, and Jink.

The control parameters are used by the rules system to select a maneuver from the maneuver list. The list of the condition parameters is unlimited, as the user can always create new parameters as needed. The most commonly used condition parameters are those that relate to the Prime Opponent (PO), typically: slant range, Angle Off Tail (AOT), Heading Crossing Angle (HCA), closure rate and relative height.

Integrating the IAT with the Ownship Simulation

An important part of the simulator build is the integration of the air combat system with the manned simulator, referred to as the ownship, and the instructor station. At all times it must be possible for the instructor to monitor the student and be able to control the engagement. If the student is slowly solving a missile tracking problem, he may well choose to leave the IAT perform its current maneuver rather than switching to a more aggressive defensive maneuvering position. Within the current system, the instructor can either select Guns Combat, Radar Missile Combat, IR Missile Combat, or any combination of all three.

The criteria used to control combat types is usually range to the Prime Opponent (PO). For example, Radar Missile Combat will be used if the PO range is beyond IR missile range and Guns Combat if the PO range is too close for an IR missile shot. In between these weapons ranges, IR Missile Combat will be used. The instructor can select either single or multiple missiles for the Missile Combat. Selection of Visual Identification will postpone the Air Combat until the student has approached the target to make visual contact.

For its in-house simulator programs, CAE has designed a comprehensive man machine interface which allows the instructor to have different levels control over the IAT, including

- selection of the type of IAT, i.e. F16, MIG-29, F4 etc.

- selection of the air-combat mode, i.e. guns, IR-missile, or Radar missile

- proficiency level, i.e whether the IAT should behave as an Expert, Intermediate, or Novice adversary

- control over such basic functions as rearm, fuel freeze, kill inhibit, and recover

- the ability to override the prime opponent rule base in order to specify the opponent

- taking control of the IAT using a rudermentary joystick and throttle at the instructor station, or control from a page input, i.e. a commanded speed, altitude, or heading

- selection of a split maneuver from high, low, spiral, left, or right

- selection the wingman mode, formation type and leader

- access to such maneuvers as Left Climb, Climb, Right Climb, Left Turn, Straight & Level, Right Turn, Left Dive, Dive, Right Dive. Target maneuvers are F-Pole, Disengagement, Pursuit and Stinger

- repositions

a) One-vs-One Engagements

One-vs-One engagements involve one fighter against another fighter. One-vs-one is usually the last situation that the pilot would like to face. However, as this may be unavoidable, it needs to be practiced in the simulator. The instructor can either set up the Ownship vs IAT (human vs computer) or IAT vs IAT (computer vs computer). The air combat modes that can be assigned to the IAT are those previously described, i.e. guns, IR-missile, or Radar-missile. Alternatively, any of the basic maneuvers available at the instructor station may be used to provide basic training in weapons deployment and disengagement.

b) Two-vs-One and One-vs-Two Engagements

Copyright © 1997 by the American Institute of Aeronautics and Astronautics, Inc. All Rights Reserved.

Two-vs-one involves a coordinated pair of aircraft against a third aircraft. Two common double attack scenarios is one where the ownship is in formation with one of the IAT's against the other IAT, and where both IAT's operate in a coordinated role against the ownship.

The double attack usually begins with the instructor selecting the formation of two aircraft. He assigns one of IAT's to be the wingman. The leader can be the ownship or the other IAT. The wingman formation selection may be double attack formation (Tactical Spread), Radar Trail, Fighting wing formation and Close Formation (Echelon, Close Trail).

c) Beyond Visual Range

Beyond Visual Range (BVR) is typical for a Radar engagement and is setup by invoking the radar-missile engagement rule base.

d) Proficiency

The instructor facility provides for a selection of proficiency levels that give the aircraft various levels of performance when maneuvering. The current system supports Expert, Intermediate and Novice. Expert aircraft will fly at the sustainable turn and use excess energy in the vertical plane. Novice will not maintain its altitude in a high "G " turn. The Intermediate level is between the Expert and the Novice. Proficiency is built into the maneuvers and the rules sets.

e) Weapons handling

Weapons handling for the IAT is dominated by the rules. The rules select the weapon type based on the current aircraft state. If the rule is satisfied, the weapon will be selected and the weapon fire request is granted. The IAT aiming module computes the aiming solution angles for the selected weapon. The aircraft is commanded to fly the profile to achieve an aiming solution and to deploy the weapon, either the bombing maneuver for the bombing run or the aiming maneuver for the gun / rockets.

Applications

CAE's ITEMS Interactive Air Combat system is a well proven technology which has matured through customer acceptance on three fixed-wing flight simulators. Under the guidance of experienced pilots knowledgeable in military flight simulator technology, CAE's scientists and engineers have made the ITEMS Air Combat system the tool of choice for pilot training in air to air combat.

The first ITEMS Air Combat system was installed several years ago and provided a guns air to air simulation with limited wingman capability and one IAT. At the time, it was a formidable challenge to design and implement such a system that could provide a high degree of realism with the ability to program any kind of air target scenario.

On the second simulator, the system was enhanced by adding new Gun's combat maneuvers and by adding the capability for the IAT to perform bombing runs against ground installations and ships. This system has also been in use for several years.

On the third simulator, the system underwent major changes with provision for missile (IR, Radar) engagements, Two-vs-one and one-vs-two engagements, and an enhanced wingman capability. To support these initiatives, a two IAT, i.e. two target, capability was built into the system. This allowed the pilot to work in concert with a computer generated counterpart. New maneuvers and rules were developed for the missile attacks and the two-vs-one and one-vs-two engagements. The two IAT capability removed the need for the control of the second target from the instructor station, or the need for a second simulator, Experience has shown the control of a second target in air combat mode via controls at the instructor station to be very difficult due to the limited field of view. In addition, the need for a second simulator is an expensive option.

During the course of these installations, several lessons have been learned. The first is the importance of the visual system and the ability to detect targets visually at acceptable ranges. Poor visual fidelity and cuing, or limited field of views, make for a poor air combat training device. Subject matter experts are essential to providing detailed information as the maneuvers performed in the various situations. The translation of these maneuvers into software is the task of the software engineer.

Figure 3 Example of Two Air Targets in Close Echelon

Copyright © 1997 by the American Institute of Aeronautics and Astronautics, Inc. All Rights Reserved.

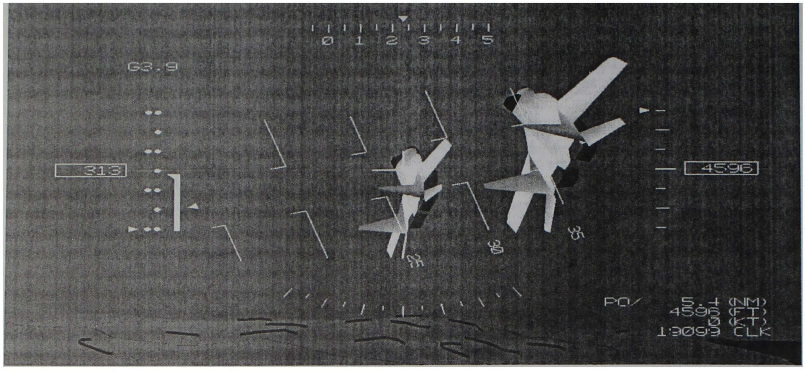


Figure 3 Example of Two Air Targets in Close Echelon Formation

Example Engagement Sequence

A sample shot from one of the air combat simulators is presented in Figure 4 which shows two IAT targets in an echelon formation executing a 4g turn. One IAT acts as the leader and the other as wingman. The head-up display is assigned to the lead aircraft and shows the "g" load, airspeed and angle of attack on the left. On the right of the figure is shown the altitude and rate of climb, with the aircraft heading shown at the top of the figure. Central to the picture is shown the pitch and roll bar which defines the attitude of the aircraft.

Summary

This paper has described the use of CAE's Interactive Tactical Environment Management System (ITEMS) for air-to-air combat training. The key features of the software package are the rule base system for scenario definition, the maneuver routines, and the six-degree of freedom air target models. The system supports training in guns, IR-missile, and Radar-missile engagements. The system caters for two air targets with a comprehensive wingman capability that provides for training in one-vs-two and two-vs-one engagements. Selection and control of the air combat is achieved through a page set-up at the Instructor Station.

Acknowledgment

Over the years, many people have contributed to the development of the air combat system. Their contributions are gratefully acknowledged, in particular Guy Beauchesne, Alick Nicholson, Reginald Decoste, Denis Mongeau, and Dave Rowe.

Copyright © 1997 by the American Institute of Aeronautics and Astronautics, Inc. All Rights Reserved.

MICROSOFT FLIGHT SIMULATOR AS A DATA CAPTURE TOOL FOR AN INTELLIGENT TUTORING SYSTEM IN PRIVATE PILOT TRAINING

Ross M. Hunter
Research Associate

Gerald P. Chubb
Associate Professor

The Ohio State University
Dept. of Aerospace Engineering, Applied Mechanics, and Aviation
164 W. 19th Avenue
Columbus, OH 43210-1110

Abstract

The FAA, NASA, and industry partners have made a call to revitalize the general aviation market. Their partnership, Advanced General Aviation Technologies Experiment (AGATE) has identified that one of the barriers to overcome is the reduction in training, both in time and cost by 50%. This reduction in training is for the period from which a student has no knowledge of airmanship, up to what we now consider a private pilot license with an instrument rating.

One way that has been suggested is the use of an Intelligent Tutoring System (ITS) for ground based, and in some instances, airborne platform based training. This would offer the ability or reducing the investment of the cost and time without reducing safety. For the ITS to gather data of the student's pilotage performance, a method of data capture in a ground based simulator is required. In this case, a low cost combination is desired in order to be both affordable to the target market and reduce the cost and time investment of instruction.

Microsoft Flight Simulator 5.1 with modification offers this type of data collection in a package which has already been deployed to market. It has the capability of simulating, with a reasonable amount of fidelity, most maneuvers required by student pilots.

By using a program running in tandem of Flight Simulator, nearly every variable associated with the aircraft flight model, the systems states, and the environment are accessible. This data can be used by the ITS to determine he student's current abilities and rate of progression.

Introduction

The FAA and NASA have developed the Advanced General Aviation Transportation Experiments (AGATE) program in an effort to revitalize General Aviation (GA) manufacturing. This program has identified one of the ways to achieve this is to reduce the training time and cost by at least 50%. A way of foster this is to apply advanced techniques in student instruction which lower the time in the aircraft and practice on the ground.

The use of Computer Aided Instruction (CAI) systems have been used in the past to teach basic aeronautical knowledge, however, an attempt to apply the instruction of pilot performance would not 'fit' under the methodologies of CAI. Intelligent Tutoring Systems (ITS), however, offer a way to teach both aeronautical knowledge and critique pilot performance because of its design to adapt to the student.

In order to supplement the ITS with pilot performance data at a low cost, a low cost, off-the-shelf flight simulation package would be required. There are several such packages on the market. Each stands out with its own individual quality, but what is most important is the ability to capture flight model data. Microsoft (MS) Flight Simulator (FS) has three qualities which make it the more attractive package. It is less expensive in comparison to other packages, is more widely used, and is offers more expendability.

A joint research effort between The Ohio State University and Systran, Inc. under a NASA sponsored SBIR contract investigated the possible use of such a system in the instruction of student pilots to the performance level and the abilities required to operate the AGATE 'type' aircraft. This effort found that it was feasible to utilize MS FS as a data collection tool for pilot performance assessment within the ITS.

Intelligent Tutoring System

Overview

An Intelligent Tutoring System (ITS) differs from a Computer Assisted Instruction system (CAI) in that the ITS does 3 things relevant to knowledge. First, the ITS must contain a model of the 'expert'. This means that the system contains records of the abilities and performance of the target 'expert'. Second, the ITS must have the ability of deducing and tracking the knowledge and abilities of the student. This is needed for the third requirement: the ITS's ability to design a path of progression of teaching the student to better match the model of 'expert' performance.[1]

This path of progression is not a "set-in-stone" instruction code, but rather an adaptive system which has the ability to tailor the instruction on a student by student basis. An ITS may also adapt to user preference, personalities, and abilities, such as their cognitive style, social style, hazardous attitudes identified by CRM courses, or even their taste in music.[2] Besides the system's ability to adapt teaching and testing to the student's preferences and model of knowledge, the system can also modify the instruction when the learning is not progressing. This prevents an end-less loop where the same subject matter is presented in the same fashion, causing the student to become de-motivated. These advantages over a CAI do come at a high cost however.

Because an ITS is not just a set of code instructions but a system that adapts to the models present, the system must first contain "... a cognitive model that is capable of solving problems in the same way we want students to solve the problems. This cognitive model is then used to interpret the student's performance ."[3] This requires that the specific subject matter be divided within the models in a form consisting of facts, rules, and inferences[4] which the computer can use to develop a particular step in a lesson, then present and test the student in a form that the user's performance can be measured and compared to previous learning. This means that because there are various skill levels at different tasks per student, there are many possible paths-of-progression. This translates to a very high amount of development time in constructing an ITS as opposed to a non adaptive CAI. This time difference can be expected to be between 3 to 10 times that of a CAI.

In order for an ITS to proceed with instruction, it must contain every ability and action that the 'expert' possesses in order to complete a specific task, no matter how subtle or mundane the action or chain of actions are. Every possible detectable cue and reaction of the 'expert' must be recorded so that it can be available to the ITS's teaching module. This level of detail requires a very close examination of how each task is completed. Along with the determination of what the 'expert' deems as a required action, alternatives must also be examined to understand why or why not the alternate action is not a the correct method to complete the task. "...(The) teaching model in an ITS consists of engaging the learner in a reasoning task, and comparing the student's actions with the set of actions that can be generated when the student does something either wrong (according to the domain knowledge) or useless in pursuit of the current task."[5] One method of achieving this is to apply the techniques of IDEF0 diagraming. This type of model forces a top-down examination of an activity and allows for a detailed description of every task and sub-task in an activity. Such a model was developed and proved to be very useful in formulating the groundwork directed toward building an ITS targeted for progressing a student to an advanced form of instrument flight.[6]

ITS Application to Flight Training

Aeronautical Knowledge

Traditional ground school training can be more easily applied to an ITS framework than that of pilot performance in that the nature of what is measured and, more importantly, how it is measured is must less abstract than pilot performance data derived from either an airborne platform or a ground based simulator. In ground school training, the level of aeronautical knowledge that the student retains after instructional presentation is the direct value to the rate and the amount of the students progress. Testing can be easily done with simple yes/no or multiple choice questions. More difficult, fill-in-the-blank questions may also be presented to the student.

Flight Performance

In the application of an ITS to the area of pilot performance, the ideal situation would be to have flight data recording from both an airborne platform and a ground-based trainer. This would allow for the ITS to adapt ground-based training to deficiencies

detected from the airborne platform. In the airborne side, data collection can be done by a specialized flight instrumentation package. This instrumentation package would utilize a Global Positioning Satelite(GPS) receiver, pressure transducers, a fixed pendulum, and a gyro connected to a low cost PC for data recording. Such an instrumentation package has been developed by Technology Systems, Inc. under a Federal Aviation Administration sponsored and Air Force monitored (SBIR) contract as Semi-Automated Flight Evaluation System (SAFES).[7] SAFES was developed from a follow-on to Basic Flight Instruction Tutoring System (BFITS) which contrasts SAFES in that it concerns with student performance for private pilot certification lessons where as SAFES extends into instrument. Although this system does a good job at recording and reporting student performance, it does not utilize the principles of intelligent tutoring.

For ground-based training, a flight simulation package must have the ability to record data regarding student piloting performance. BFITS has this ability, however there are several drawbacks to using it. First, the visual scene simulation does not provide local scenery. Local scenery aids in cross-country training and local area pilotage. Second, because of the specialization of the software and it's low number of copies in the overall market, it is very expensive relative to other flight simulation packages.

Microsoft's Flight Simulator in contrast is much more affordable largely due to it's high volume deployment to the home market. It also offers the ability to add and/or modify the objects in the visual scene. MS FS is deficient in a few respects such as unrealistic simulation of radio navigational aids where other packages such as IFT-Pro, Jeppesen's FS-200 and AzureSoft's Elite do more accurately. However, the advantages of a larger support base and the ability to make modifications with 'plug-ins' make MS FS a better choice. One of these 'plug-ins' is a type of Flight Simulator Object Driver (FSO) that allows from the collection and recording of variables within MS FS.

In both cases, airborne or ground based, the results of testing are contained in the telemetry data recorded from a particular flight exercise. The data recorded for each exercise must be compared against a template or set of templates which describe the perfect execution for the tested exercise. Accompanying each template is a threshold description. This serves several functions. First, a pass/no-pass determination can be made. If the data recorded from the exercise 'fits' within the template's threshold, then the exercise has been performed successfully. Second, the student's performance in the task can measured, regardless of the pass/no-pass determination. Finally, rate of progression can be determined by comparing the results of the exercise against previous recordings.

Microsoft Flight Simulator

History

Flight Simulator has evolved as newer and faster computers for the home market have been introduced. From the beginning in 1979 when Bruce Artwick introduced Flight Simulator for the Apple II, visual and flight realism has been a challenge. The typical home computer at that time consisted of only 16k of memory, BASIC language capable of only integer computations, a digital storage system consisting of a cassette player/recorder as the main component, and display circuitry capable of only 3 colors.

The next step in the growth was introduced in 1983 as Flight Simulator II after a 9 month development. This version took advantage of the Apple II's ever increasing capabilities which included 48k of program memory and 16k of screen memory. Other enhancements included the move from black and white line graphics to solid colors and the expansion of the 'gaming area' from a few hundred square miles, to more than 100,000,000 square miles of coverage. This increased coverage allowed for flight over the entire United States. The last major improvement to note was the more refined flight model which was more difficult to manipulate then the previous, but again, was more realistic.

Subsequent versions of the software were improved to take advantage of the capabilities that were being introduced to the home desktop market. The simulator moved to the Apple Macintosh platform from the pervious 6502 Apple II system and a companion version was introduced for the IBM platform which was becoming accepted as a machine for home use based on its new afordability. Version 4.0 saw the last release of the simulator from the Macintosh platform as the next releases were focused entirely on the IBM PC compatible market which enjoyed extreme user acceptance over that of the Macintosh due to affordability.

User Acceptance

Several factors accounted for the popularity of Microsoft Flight Simulator (MS FS) in the home market. First, there are vast numbers of IBM PC platforms on which the current versions operate. Second, and more importantly, MS FS can to be 'modified' by the user. Utilizing simple 'plug-ins', the program can be upgraded, either in terms of scenery or in terms of aircraft handling qualities and a more complex and detailed visual rendering of the earth. This can be done by creating a scenery file developed by the end user with either off-the-shelf or shareware software obtainable over the internet. Another 'plug-in' is the aircraft file. This allows the user to add more types of aircraft to the MS FS selections list. As with the scenery files, these aircraft files are developed using off-the-shelf software, and range from the basic Cessna 152 up to the latest Boeing 777, including many types of military aircraft. What is so appealing about these 'plug-ins' is the fact that they are for the most part developed by the end users and can be distributed amongst these enthusiasts via the internet at no cost for the 'plug-in' itself.

The ability provided by the internet for the end-users to share their work has fostered several file and web servers all over the world to serve as repositories of everything that has been developed. Although the current version of MS FS is version 6.0 (Flight Simulator for Windows 95), previous versions back to v4.0 are still supported by the Flight Simulator enthusiasts. To see the vastness of this support one needs only use a web browser and search for "Flight Simulator". The number of returns would take days to visit and scan each one.

Another thing to mention that has been shared among these users are adventure files. Adventure files are situations where the combination of environmental conditions, the flight objective, and the type of aircraft in the adventure can be distributed and run by many users. Adventure files challenge simulation pilots to conditions that are not part of basic cross-country flight. Every simulator pilot which uses an adventure starts off with the same set of situation parameters as the original user who constructed it. These situational conditions include the weather and climatic changes, the aircraft being used, and preprogramming system faults, if any.

Current Version

The current version, 6.0, operates within the Windows 95 Operating System as opposed to previous versions which ran within the DOS or Windows 3.1 environment. With a few exceptions, everything that has been developed for the version 5.1 runs on the 6.0 version. Microsoft developed this new version to take advantage of 32-bit processing in contrast to 16-bit processing in the previous 5.0 version and allow clean operation within the Windows operating system which had been anything but suitable previously.

FLIGHT RECORDER

Operation

As stated before, the addition of 'plug-ins' to extend the capabilities of the program. The means by doing this through a file type FSO, or Flight Simulator Object driver file. Microsoft Flight Simulator is aware of the presence of these FSOs by declaration lines within it's .INI or initialization file. This INI file is much the same as the AUTOEXEC.BAT or WIN.INI file familiar to users of DOS and Microsoft Windows. When MS FS is launched it looks within the INI file to run under the correct settings and know which files to load into it's memory heap to execute along side of the main program itself. Flight Simulator then issues event calls to the FSOs regarding the current status of the program.

The purpose of this particular FSO that we were interested in is to record selected variables residing in the memory heap of MS FS. It's first job, once executed, is to create and open a data file in the file system in which to copy the information to. If this action is successful, then each time a Frame Event or Second Event is issued by the simulator, whichever is being watched for by the FSO's module, the FSO will check to see if all conditions are met required for a recording. Once a recording procedure is called, the variables desired for recording are each copied into an internal memory string then copied into the open data file. Once the MS FS program halts, the FSO begins the Exit Procedure call in which it will close the created data file.

Accessible Variables

Due to the large number of MS FS enthusiasts and the lines of communication that the internet fosters, these users have been able to network and work together to share each one's personal insight into the operation of the software. One of the results of this has been a large number of documented internal variables residing in the memory heap of MS FS. There are over 350 known variables documented including the record staring byte location, the variable offset length, and the description of the variable. These variables describe the numerics for both the display and flight model engines.

Starting Byte	Record Length	Description
02EE	2	Ground elevation
02F4	4	Magnetic north deviation
02F6	2	Inner marker flag
02F8	2	Outer marker flag
02FA	2	Middle marker flag
0714	2	Crash damage flag
0716	2	Offrwy crash flag
0788	2	Gyro deviation
07BC	2	ADF frequency
07BE	2	COM1 frequency
07C0	2	COM2 frequency
0864	3	True airspeed
0867	3	Indicated airspeed
0864	3	Vertical airspeed
08E1	5	DME1 reading
0A34	2	Flap angle
0A38	2	Flap 1
0A3A	2	Flap 2
0A72	1	COM operational
0A73	1	NAV operational
0A8C	2	Elevator position
0A8E	2	Aileron position
0A90	2	Rudder position
0AA3	2	Elevator indicator position
0AA5	2	Aileron indicator position
0AA7	2	Rudder indicator position

Table 1- Selected Accessable Variables

FLTREC

Released in December of 1995, FLTREC allowed for recording of data in one second intervals. This data included the elapsed time, time of day, latitude, longitude, altitude, pitch, bank, heading, true air speed, and gear position. To use this FSO, the user needs to first tell MS FS to run this program segment by including in within the flight simulation's INI file. Then, while the user is flying the simulation, the user sets the transponder to 0001 to begin recording. To stop recording the user would set the transponder to a value other than 0001. While recording, the FSO would write the data in binary format into a file named fltrec.dat. In order to convert the file into something legible, the user must run a translation program outside of flight sim. Both the FSO and the translation program were written by Adam Szofran who has also written several other programs which interface with MS FS and are available free of charge via the internet.

Our Implementation

Because FLTREC did not record all of the variables that we were concerned with, we had to develop and test our own version to validate this approach. To do this, several steps were required. First, we identified the crucial abilities that we required of the data capture FSO. These abilities included the ability to record data at variable times or at different 'frames', the ability to record any of the known variables within the memory, and the ability to 'decide' to record a variable based on a predefined threshold. Second was the development of the instruction code. This was done in assembly-language and written by 2 visiting internship students as part of their field experience required by their college. After a successful compile of the assembly code, it was subjected to another compiler, FASAM (Freeware Scenery Assembler for Microsoft Flight Simulator), written by Adam Szofran. This compiler took the compiled binary and converted it to a type FSO, loadable into the memory of MS FS. The resulting file was applied to Flight Simulator as tested giving results similar to FLTREC, but with the differences that we required. Data samples were taken on every 'frame' or each time MS FS completed a redraw of the display, as opposed to every second as did FLTREC. Collected data included the variables of concern which met the requirements set in their respective thresholds.

The platform used for testing was a Pentium 66Mhz with 8 Megabytes of memory and an STB

Express video card. The additional instructions of the FSO captured the data without any noticeable degradation in the performance of MS FS. This proved that this was a viable approach to collect pilot performance data from a low cost, off-the-shelf flight simulator.

Summary and Conclusions

The Intelligent Tutoring System offers the ability to better engage the student in learning. This type of system can assist in teaching and critique pilot performance as long as data regarding the flight as available to the system. Airborne systems and ground based, hi-fidelity simulators can currently offer this data, however at a very high cost to the student.

Microsoft Flight Simulator is currently able to offer the fidelity required to teach some aspects of private pilot instruction. With the addition of more detailed local scenery, the simulator can become effective as a tool to help teach pilotage. Due to the deficiency in MS FS to accurately simulate instrument navigation, it would require additional re-work and/or a 'plug-in' to allow for effective instrument training.

Utilizing techniques to capture data from an off-the-shelf flight simulator to aid in the use of an ITS can bring flight instruction into the home. The ability of an ITS to tailor instruction on a student-by-student basis can further reduce the time and cost involved in producing a proficient pilot. This combination would aid in increasing interest in general aviation and revive a decreasing industry.

---

[1]Burns, Hugh L. and Charles G. Capps (1988), "Foundations of Intelligent Tutoring Systems: An Introduction," in M.C. Polson, J.J. Richardson, and E. Soloway, editors, Foundations of Intelligent Tutoring Systems, Lawence Erlbaum Associates, Inc., Hillsdale, NJ, 1-19.

[2]Chubb, Gerald P., Ross M. Hunter and Michael Motsis (1996), Intelligent Tutoring Systems for AGATE Training, NAS1-20517 Phase I STTR, Final Technical Report, The Ohio State University, Columbus, OH, & Systran, Inc., Dayton, OH.

[3]Anderson, John R., Albert T. Corbett, Jon M. Fincham, Donn Hoffman, and Ray Pelletier (1992), "General Principles for an Intelligent Tutoring Architecture," in egian and Shute, editor, Cognitive Approches to Automated Instuction, Lawence Erlbaum Associates, Inc., Hillsdale, NJ, 81-106. p. 82

[4]Yazdani, Masoud (1987), "Intelligent Tutoring Systems: An Overview," in Lawler, Robert W. and Masoud Yazdani, editors, Artificial Intelligence and Education, Volume One, Learning Environments and Tutoring Systems, Ablex Publishing, Norwood, NJ, p. 182-201.

[5]Larkin, Jill H. and Ruth W. Chabay, "Introduction," in Larkin, Jill H. and Ruth W. Chabay, editors (1992), Computer-Assisted Instruction and Intelligent Tutoring Systems: Shared Goals and Complementary Approches, Lawence Erlbaum and Associates, Hillsdale, NJ, 1-9.

[6]Chubb, Gerald P., Ross M. Hunter and Michael Motsis (1996), Intelligent Tutoring Systems for AGATE Training, NAS1-20517 Phase I STTR, Final Technical Report, The Ohio State University, Columbus, OH, & Systran, Inc., Dayton, OH.

[7]Benton, Charles J., Paul Corriveau, Jefferson M. Koonce, and William C. Tirre (1992), Development of the Basic Flight Instruction Tutoring System (BFITS), AL-TR-1991-0060, Human Resources Directorate, Manpower and Personnel Research Division, Brooks AFB, TX.

AIAA-97-3516

MICROSOFT FLIGHT SIMULATOR SUITABILITY FOR CROSS COUNTRY EXERCISES FOR PRIVATE PILOT TRAINING

Gerald P. Chubb
Associate Professor

Philip M. Macy
Honors Student

The Ohio State University
Dept. of Aerospace Engineering, Applied Mechanics, and Aviation
164 W. 19th Avenue
Columbus, OH 43210-1110

Abstract

All aviation students at OSU take the private pilot's ground school (AV 310). Aircraft Systems majors then take two flight lab courses. The first (AV 341) gets them to solo flight; the second addresses cross country flying (AV 342). Aviation management and human factors majors typically do not take either course. The latter must take the academic ground school for the instrument rating, and many have difficulty due to no flight experience.

AV 342 was reviewed to determine the imposed requirements. MS FS exercises were used to emulate tasks required in the syllabus. MS FS has the capability to simulate all required flight maneuvers, but it does lack fidelity in the area of aircraft handling qualities. It is therefore not a substitute for the aircraft in learning the psychomotor skills for safe flight. MS FS also has an unrealistic representation of radio navigational aids, which are always available. Two additional capabilities are desirable: a) radio communication between pilot and air traffic control specialists, and b) simulation of local area scenery. MS FS has capabilities for cognitive task training and procedures rehearsal that are impressive for its cost. Properly used, MS FS is useful, but it is not sufficient "right out of the box."

Introduction

The Federal Aviation Administration (FAA) has recently approved the use of Personal Computer (PC) Aircrew Training Devices (ATDs) in formal flight instruction. Prior studies by Hampton et al.[1] demonstrated the utility of PC ATDs in instrument flight training. Several different vendors offer PC ATDs of varying capability. Among these are ELITE, IFT-Pro, and FS-200. MicroSoft (MS) Flight Simulator (FS) was originally regarded as "game" software and was often not seriously considered as a possible PC ATD by pilots. However, it has two distinct advantages: a) it is inexpensive, and b) it is widely used among non-pilots. While there is now a Windows 95 version of MS FS 6.0, the study reported here is based upon a Senior Honors Project[2] and reviews MS FS 5.0. The work had been planned and started as early as MS FS 4.0.

The Ohio State University (OSU) operates a flight education program under Part 141 of the Federal Aviation Regulations (FARs). The academic program for aviation majors includes three options or areas of concentration (AOC): a) aircraft systems, b) aviation management, and c) human factors.

The aircraft systems majors are pursuing a career in professional aviation, leading eventually to an Air Transport Pilot's (ATP) license. All students in the aviation program are required to take the academic ground school for the private pilot's license, but aviation management and human factors students do not complete the flight laboratories required to get the private pilot's certificate.

However, students in the human factors AOC are also expected to complete the academic ground

Copyright © 1997 by the American Institute of Aeronautics and Astronautics, Inc. All rights reserved.

school for the instrument rating. Without any flight experience, these students find it difficult to deal with the topic content of this course.

It is unreasonable to demand that human factors majors complete flight training, due to the cost. However, it would be desirable to emulate that experience in a low-cost flight simulation laboratory. Were that laboratory available, aviation management students could also be offered the opportunity to develop a better appreciation for the operational environment of flight, and aircraft systems majors could perform pre-flight rehearsals.

Preliminaries

Out of the box, MS FS is not well-suited to flight instruction: pilots do not fly keyboards. The Virtual Pilot control yoke from CH Products provides a valuable addition and is priced at about the same cost as the MS FS software. MS FS is designed to provide coordinated turns for the novice user, but this provides more help than desired.

To train prospective pilots properly, it is therefore necessary to add a rudder pedal kit. Neither of the yoke or the rudder pedal controls will have force reflection feedback, so students will not gain an appreciation for the handling qualities of the aircraft, nor will they learn how these handling qualities change with atmospheric conditions. Flight experience is still required to learn how atmospheric conditions affect the handling characteristics of the aircraft.

Since the rudder pedals have to absorb considerable force, a somewhat larger investment in a more rugged kit is well-advised, about doubling the cost so far (if Thrustmaster's rudder pedal kit is used). Even so, the total cost is still less than using one of the higher priced software products (ELITE, IFT-Pro, or FS-200). Moreover, we have found a number of third party software products (e.g., Aircraft Adventure Factory, and Apollo's Scenery and Object Designer) available for MS FS 5.0 that are not available for these other products.

Background

The MS FS software does allow setting up datasets that establish the operating conditions for a particular simulation exercise. This allows some tailoring of the flight experience that a user will encounter during the exercise of MS FS. The basic choices of location, aircraft type, weather, and computer screen configuration are saved for later recall as a specific flight MODE (called a "Situation" by MS FS 5.0).

This MODE file is used at program initiation, and after a crash, should one occur, to control run conditions. Specific lab exercises (e.g., normal versus crosswind landings) require the generation of special flight MODEs to demonstrate specific maneuvers.

Demonstration exercises (DEMOs) in MS FS are different from MODE files. They can be used to show a student what a maneuver should look like, but they must be created and replayed by using keyboard commands. They are like a "canned" mini-scenario.

Capabilities Assessment

While OSU flies Cessna 152s, MS FS only has the Cessna 182 (C-182), a more capable aircraft. It best emulates a C-152 by being flown with the gear down (although the C-182 has retractable gear, the C-152 does not). Also, the C-182 has an adjustable pitch propeller and a manifold pressure gauge, neither of which are shown on the MS FS 5.0 displays, but the C-152 does not have this capability, so the MS FS displays for the C-182 are actually an adequate representation of the C-152 cockpit.

MS FS does have some built-in features to measure and display flight path records. Instant replays of the last 49 seconds of flight are always available. Landing Analysis will display a graph of the aircraft's vertical speed profile over the final 200 feet of descent prior to touchdown, and Maneuver Analysis will display a graph of the ground track following any maneuver.

Also, as explained in a separate paper, it is possible to collect data on real-time operation of the program, over and above the data captured MS FS itself. This led us to believe that MS FS might be more useful as an instructional aid than

previously believed, especially by those who simply saw it as a computer game. Performance measures could be provided to student and instructor that would assess the pilot's proficiency in accomplishing various flight maneuvers.

In order to determine the suitability of MS FS 5.0 to support private pilot flight instruction, the existing Flight Syllabi for AV 341 and AV 342 were examined. AV 341 is the preliminary dual (student and instructor) flight instruction course that takes the student to the point where solo flight is completed. AV 342 then addresses the cross country requirements that must be completed to get the private pilot's certificate. At this stage, the dual instruction is more limited, and the student is more often able to fly alone in completing flight laboratory exercises.

Table 1 summarizes the tasks which cannot be done by MS FS that are covered in OSU's AV 310, the Academic Ground School for Private Pilots, which precedes AV 341 and AV 342. Table 2 summarizes additional tasks, not covered in AV 310 which are also not supportable by MS FS. The flight maneuvers required in each course were reviewed to determine whether they could be accomplished successfully using MS FS 5.0. While all of the required maneuvers could be done, several deficiencies were noted.

Table 1. Unsupported AV 310 Tasks

Illustrate Aircraft Performance Factors
Perform Direction Finding (DF) Steering
PIREPS / WX Updates
Simulated Instrument Radio Aids / Radar Services
Pilotage (Day / Night)
Controlled / Uncontrolled Airport Ops.
"Lost" Procedures

MS FS Deficiencies

MS FS is not equipped to support instruction on pre-flight inspection of the aircraft. The engine start procedures are not realistic either. MS FS 4.0 did not support taxiing instruction, since there was no provisions for toe brakes, but 5.0 does allow toe brake inputs. Some rudder pedal kits do include toe brakes now for MS FS 5.x and the more sophisticated PC ATD software products.

Table 2. Other Unsupportable Tasks.

ASR Approach
Emergency Descents / Climbs Using Radio Aids / Radar Vectors
Standard Departure Procedure
Practice Area Orientation
Standard Arrival Procedure
Engine Start / Shutdown
Cross-Controlled Stalls
Rectangular Pattern Tracking
Rectangular Pattern with Airspeed Transitions
Forward & Side Slips
S-Turns Across a Road
Slow Flight for Traffic Separation
Fuel Management
Emergency Procedures
Divert to Alternate
Black-out Night Landing

Seven specific MS FS deficiencies were noted for flying operations: a) views, b) communication, c) local scenery, d) lack of magnetic variation, e) no altimeter adjustment ability, and f) navigation aid availability, identification, and distance measurement. These deficiencies are not uncorrectable but need to be dealt with if the package is to be used successfully for flight training. However, MS FS does offer some views of the simulated flight that can actually aid the instructor in being able to illustrate or explain certain flight phenomena.

View Deficiencies

The computer screen can be totally dedicated to the "out-the-windscreen" view or it can be split into two parts, where the lower portion gives the simulated cockpit instrument display. The upper portion in that case can still display any one of eight views, from straight ahead to straight behind, in 45 degree increments.

The out-the-windscreen view can be modified but is always restricted. This complicates approach and landing procedures training, since it is desirable to look off-axis, out the side window. Using MS FS 5.0, changing the view requires

several keystrokes, and the image is always presented in front of the pilot, even though it emulates the view out the side window of the aircraft.

There are solutions to this problem, but they are not quick, cheap, or simple. By developing special software and adding some hardware, the FAA's Civil Aero-medical Institute (CAMI) in Oklahoma City has gotten a wider field of view by using multiple computer screens.

View Advantages

It might be noted that MS FS offers some additional views which have educational value. One can get a side view of the aircraft one is flying, which allows a student to monitor and watch what happens when the aircraft is put through a stall maneuver. This is helpful in both explaining what happens in a stall and preparing the student for that experience.

It is also possible to add an inserted window that views the simulated aircraft from a "spot" plane, as if you were flying in formation and could obtain a telemetered video from a wingman. In this way, the student pilot can watch their own aircraft pitch up and slow down while also monitoring these changes on the simulated cockpit instruments. This would be difficult and expensive to do in actual flight training!

Communication Deficiencies

There is no radio communication or transponder capability in MS FS 5.0. Getting appropriate clearances, enroute weather, and Automated Terminal Information System (ATIS) data all require radio operation. A transponder allows an ATCS to know the aircraft's tail number, heading, speed, and altitude. In some instances, an ATCS may ask the pilot to change the "squawk" frequency on the transponder, which MS FS cannot do. In an emergency, the pilot is supposed set the transponder to 7700, and if two-way radio is "lost" the transponder is set to 7600. MS FS can do neither.

When a communication capability supple-ments use of MS FS 5.0, a much more realistic flight operating environment can be created. That requires a headset and intercom system, as well as appropriately trained surrogates for the various Air Traffic Control Specialist (ATCS) personnel that would be involved.

Scenery Deficiencies

While very detailed scenery simulations are available for certain metropolitan areas, local area scenery is not typically available. Basic runway simulations are available on the internet, but surrounds in the vicinity of the airport are not. Third party software is now available for creating these scene simulations (e.g., Apollo's Scenery and Object Designer for Microsoft Flight Simulator 5.x), but creating such scene simulations is still a labor intensive effort.

Lack of Magnetic Variation

The standby compass in an aircraft points to magnetic North, which is NOT the same as true North. The variation between true and magnetic heading indications depends o the local magnetic variation, which is not constant but depends on one's location. Also, in the aircraft, one has to align the directional gyro with the magnetic compass indication, and as they differ, make adjustments. In MS FS, the directional gyro and standby compass ALWAYS agree, something that would not occur in a real aircraft, This eliminates an element of in-flight workload: making the adjustments.

No Altimeter Adjustments

Absolute altitude indications are based upon pressure sensing, since atmospheric pressure decreases as altitude increases. However, barometric pressure is not constant. On any one day, it varies from one locale to another. Pilot's therefore need to enter a setting into the device that corrects for local atmospheric pressure, and this value should be periodically updated on longer (e.g. cross country) flights. MS FS does not provide for doing so, again eliminating an element of pilot workload.

Nav Aid Deficiencies

Three deficiencies were noted: a) availability, b) identification, and c) distance measurement.

Availability. Very high frequency Omni-Range (VOR) stations are used as navigational aids during Instrument Flight Rule (IFR) operations. VORs are also convenient fixpoints along a route of flight, even when operating under Visual Flight Rules (VFR). VORs are line-of-sight devices and have limited range (about 70 nautical miles, nominally). Normally, one cannot detect a distant VOR's signal until airborne. In MS FS 5.0, VORs are always available, which is an advantage pilots do not have.

The more expensive flight simulation software packages (e.g. IFT-Pro) do attenuate VOR signals more realistically and are therefore preferred for IFR operations training. MS FS has generally superior visual scene capabilities, which makes it more appropriate for VFR operations.

Identification. VOR stations can be identified by their three letter identifiers, which are broadcast in Morse code. MS FS does not generate these Morse code identifiers. The user must simply assume that the right VOR frequency has been entered and that the signal being received is from that VOR.

Distance Measurement. Since VORs generate line of sight signals, the distance measuring equipment (DME) reports slant range to the station, not ground range. MS FS reports ground range, not slant range. For low altitudes and short ranges, the differences are nominal, but for longer distances and higher operating altitudes, the differences grow larger, and therefore MS FS is a less realistic representation of actual flight operations.

One somewhat obvious and perhaps critical consequence of this difference is the indicated distance at VOR crossing. In-flight, this distance would be non-zero and an indication of aircraft altitude above ground level (AGL). In MS FS, this value is zero at station crossing, giving no indication of height above ground.

Flight Exercises

The flight lesson support requires more than just the dataset for controlling MS FS. It also includes the preparation of appropriate weather reports and charts as well as sectional maps to support the planned route of flight. Consequently, each lesson assumes a scenario script has been prepared, from which the other materials can then be derived. It was also necessary to prepare a lab exercise manual for the student's use in completing the assignments. This manual not only gives instructions for each lesson, it also asks questions about the lesson.

Students are expected to have an E6B computer, a plotter, and three sectionals: Cincinnati, Detroit, and Chicago. They will also have to have the Airport Facilities Directory. For students desiring an accompanying textbook, Garrison[3] might be recommended, although OSU typically uses the Jeppeson[4,5] series. Williams[6] is an alternate text one might try, and Haines and Flatau[7] would serve well with any of these.

OSU still operates on the quarter system, rather than semesters, so the course of instruction is limited to a ten week period. Classes are held in two sessions per week: Mon.-Wed., Tue.-Thurs., or Wed.-Fri. Typically, flight lessons are of about an hour's duration, with additional briefing and debriefing time before and after each flight.

Since the intent was to give human factors and aviation management students a lab-oratory equivalent to the flight labs, in order to heighten their appreciation for the flight environment, it was decided to have only eighteen lessons, not twenty. The extra two lessons could then be dedicated to actual dual instruction flights, in order to assure the student acquired an appreciation for the differences between actual and simulated flight, including the force reflection feedback from the actual vehicle handling qualities. Students would not takeoff or land by themselves, but they would be permitted to execute coordinated turns, altitude change procedures, selected maneuvers (e.g. turns about a point and S-turns along a line: road or rail track), and would be shown both slow flight and the stall series.

Lesson 1

This lesson introduced basic VOR navigation under no-wind conditions. The session begins with the aircraft flying at 5500 feet above Mean Sea Level (MSL) about twenty miles from the Mansfield (MFD) VOR on the 220 degree radial. Inbound to the VOR, the aircraft heading will be 40 degrees, since the radial always points away from the station.

The first phase would have the student approach the MFD VOR station, keeping the course deviation indicator (CDI) needle centered, note the Distance Measuring Equipment (DME) readout decrease on approaching the station, and watch the TO/FROM flag reverse when passing the VOR.

During the flight into the VOR, the student would be directed to deviate off-course, observe what happened to the CDI, and to turn back onto a heading of 40 degrees, noting that the CDI is NOT centered, because the plane is now flying a course parallel to the desired radial. To intercept the 40 degree radial, the student is directed to turn 30 degrees to the right until the CDI is approximately centered, and then roll out on the desired 40 degree heading, trying not to overshoot. Subsequent small corrections may be needed to keep the CDI centered.

The student would then be directed to intercept a different radial outbound from the station. Once that was accomplished, the student would be shown "reverse sensing" by rotating the OmniBearing Selector (OBS) knob 180 degrees, which is accomplished with MS FS by using the mouse.

Turning 180 degrees, the student would then be asked to find the radial on which the aircraft is now flying, select the appropriate OBS setting, and fly to the VOR. The student would then be asked to deviate about 10 degrees from course and then re-intercept the radial.

Finally, the aircraft would be flown without intentionally keeping track of present position, in order to set up the determination of position by using 2 VOR stations to take a position fix. The exercise ends by flying back to the designated airport and landing.

Lesson 2

This lesson repeats lesson 1 but uses a high-wind flight mode to teach compensation under a cross-wind condition. The student must fly outbound from the VOR with a 65 degree cross-wind. Then the OBS selector would be rotated another 20 degrees, and the student would have to track the new course, with a slightly altered cross wind. Finally, the student would reverse course and fly back to the VOR, correcting in the opposite direction.

Lessons 3

This lesson again repeats the paired no-wind / wind cases for Automatic Direction Finding (ADF) indications from a non-directional beacon (NDB) navigation aid. The student flies to the station, passes it, and then flies outbound, keeping the ADF needle centered. The procedure is then repeated with a 90 degree cross-wind. The MS FS flight path analysis feature is then used to review how well the student had maintained a straight ground track.

The rest of this lesson is done without MS FS. The student reviews cross country flight planning and prepares for the next flight lesson by plotting a course from the university airport to Mansfield and filling out a flight log: distance and Expected Time Enroute (ETE) between checkpoints and heading (adjusted for forecast winds, magnetic variation, and compass deviation).

Lesson 4

This lesson begins with a takeoff from OSU's runway 27 L (left, the southern most of two parallel runways oriented East-West), presuming prevailing light wind conditions -- from the west or southwest. The student is expected to fly in accordance with the established local departure procedures, which should have been learned in AV 341. The Mansfield (MFD) airport is Northeast of Columbus and within an hour's flight.

The flight will require that the student locate and identify pre-selected checkpoints (which will need to be incorporated into the local area scenery, when created). The student is expected

to then log the time of checkpoint passage and complete other elements of the flight log: computing ground speed and adjusting Expected Time of Arrival (ETA) at successive checkpoints.

Here the student must also make note of variations in the wind correction angle and ground speed in each direction of flight (to and from Mansfield). It might be noted that the instructional value of this lesson is less dependent upon the aerodynamic fidelity of the flight simulation itself than it is on the fidelity of the cognitive workload imposed upon the student pilot performing these calculations.

Lesson 5

For this lesson, the flight to Mansfield is repeated, only this time the actual winds are different from the predicted winds. The student's calculations in-flight will therefore be quite different from the previous calculations done for flight planning, and so the flight plan cannot be used as a reasonableness test on the results of those calculations -- as it could be for the prior lesson.

Lessons 6 and 7

This set of lessons involves a flight from OSU to Sandusky (SKY), which is further North and permits utilization of the Mansfield VOR. Again, the actual wind will be slightly different than predicted. Headings will therefore be different than planned, and the VORs will help determine the new headings. With the different wind, the groundspeeds will also be different and all ETAs will need to be re-computed. One other variation is introduced: the ceiling at Sandusky will be lower than planned, requiring a change in the planned altitude to maintain the FAR mandated separation distance from clouds. MS FS permits programming of the cloud layer.

Lessons 8-9

In this case, the destination airport is Zanesville (ZZV), almost directly East of the university airport. While winds and ceiling will be "as forecast," the student will be required to divert to Cambridge and land. This will require selection of an appropriate radial to fly from Zanesville to Cambridge. The student will then be required to replan the flight back to the OSU airport, which can be done by either flying direct or flying back to Zanesville and then to OSU. In either case, the student will have to fill out the flight log appropriately for the planned route of flight.

Lessons 10-11

In this lesson, the student will fly to Lunken (LUK) airport, near Cincinnati, South of Columbus. The winds this time will be different than forecast, and the pilot will be required to correct for winds based on pilotage, without reference to the VOR. This requires identification of each checkpoint along the planned route of flight. If the pilot is not attentive to checkpoint identification, it is quite possible to get "lost" -- in which case careful map study and correlation with ground objects will be needed. This clearly places demands on the fidelity of scene simulation in order to accomplish the desired training.

Lessons 12-13

This trip will be to Northwest Ohio, with the planned destination being the Van Wert (VNW) airport. The student will have to use the NDB to navigate over a long distance without using visual checkpoints.

Lessons 14-16

Having now completed several cross country flights in different directions, under different wind conditions, with different navigational techniques, the student now plans a multi-segment, long distance flight, plotting the route on two different sectional maps.

This flight will go from OSU to Muncie, IN (MIE), to Toledo Express (TOL), and back to OSU. FAR Part 61 requires that a private pilot be able to complete a flight of 300 KM. This flight satisfies that requirement. The weather will diverge over time from what was forecast. Ceilings will be lower, requiring a lower flight altitude. Both wind speed changes and gusting will be introduced.

Since landings are assumed to occur at each of the three airports, this exercise can be completed as three separate flight modes, one from each of the three airports. Working with two sectional maps will complicate the flight planning process and flight following as well. This lesson's difficulty is not strictly determined by the simulation software. It is governed by the design of the scenario script!

Lessons 17-18

This lesson will be an end of course exam session. The student will be required to plan and then fly one-way to Fort Wayne, IN airport (FWA). Again, this will require two sectionals and the actual weather will be different from that which was forecast.

Summary and Conclusions

With the inclusion of additional interface devices and communications equipment, MS FS 5.0 can be converted into a reasonable training aid for private pilot instruction. The key is in devising good scenarios and providing the necessary supplementary information (weather materials, charts, etc.), not just relying on MS FS capabilities.

Because of the inherent limitations in a single screen visual scene simulation, MS FS is not particularly well suited to performing maneuvers about points or lines, so doing pattern work and practicing approaches is more difficult than in the aircraft. Also, because of the lack of magnetic variation, barometric pressure adjustments, and transponder setting changes, the enroute workload is lower than in the real aircraft.

To be really effective, there is a need to develop local area scenery, something which will take time and effort to accomplish but does not require frequent updating once in-place. For instrument flight, other products may be more appropriate, since they remove some of the navigational aid deficiencies of MS FS 5.0

Because MS FS 5.0 is so inexpensive, it could become a reasonable platform for developing safety seminar training material and general aviation promotional material. Many more people can afford to buy and use this package than the other more capable packages.

As long as the emphasis is on teaching the more cognitive aspects of flying (planning, flight monitoring, navigation, judgment, decision making, etc.), the impact of MS FS deficiencies may be minimized. Much more can be done with this software than is apparent from either product advertising literature or the casual critics' remarks.

References

1. Hampton, Steven, Willam F. Moroney, Tom Kirton, and David W. Biers (1993), "A Tool for Evaluating Student-Pilot Performance During Instrument Flight Training", 7th International Symposium on Aviation Psychology, OSU, Columbus, OH, 934-938.

2. Macy, Philip M. (1995), Using Commercial Simulation Software for Ab Initio Flight Education, A Senior Honors Thesis, The Ohio State University, Columbus, OH.

3. Garrison, Paul (1991), Cross Country Flying, 3rd edition, TAB Books, Blue Ridge Summit, PA.

4. Jeppesen Sanderson (1995), Aviation Fundamentals, seventh edition, Jeppesen Sanderson Training Products, Englewood, CO.

5. Jeppesen Sanderson (1994), Instrument / Commercial Manual, ninth edition, Jeppesen Sanderson Training Products, Englewood, CO.

6. Williams, J. R. (1991), The Art of Instrument Flying, 2nd edition, TAB Books, Blue Ridge Summit, PA.

7. Haines, Richard F. and Courtney L. Flatau (1992), Night Flying, TAB Books, Blue Ridge Summit, PA.

AIAA-97-3517

VIRTUAL REALITY ENTERTAINMENT TECHNOLOGY APPLIED TO MILITARY PARACHUTE SIMULATION FLIGHT TRAINING

Jeffrey R. Hogue*
R. Wade Allen†
Systems Technology, Inc.
Hawthorne, CA 90250

Ross Q. Smith‡
3Dfx Interactive, Inc.
San Jose, CA 95134

Abstract

Recently low-cost, high-performance technology has been developed to bring PC and arcade gamers the high level of visual realism found in extremely costly military and civilian professional training systems. This technology has been applied to allow the military to safely teach, plan and practice dangerous parachuting missions. A parachute flight simulation system is described which has been developed, installed and is now in use where these improved display devices are used to provide needed scene cues and extend training syllabus scope. The relationship of the technology capabilities to training requirements is detailed.

Introduction

Parachute simulation training was originally developed for operational jumpers such as smokejumpers [1], paratroops, and special forces [2]. These systems are now finding a finding a place in training emergency aircrew [3 4 5], for whom this topic is particularly critical, as they have none of the operational jumpers' [6 7] options on critical mission factors such as landing terrain, wind maximums, hostile location, and time of day or night. Justifications for simulator flight training are straight forward: safety, availability, economy, and efficiency. Military and commercial aviation communities regard simulator training as standard and essential methodology, despite very high costs of acquisition and operation to meet specification for display detail.

Parachute Flight Training Requirements

Parachutes allow rapid deployment of operational mission specialists when fixed or rotary wing aircraft alone are inappropriate. However, parachutists who are injured on landing, or land in the wrong location, degrade or nullify operational effectiveness. Skillful parachuting is vital to mission success.

Teaching parachute flight has always been a difficult task. Safe, accurate parachuting requires both training and the practice of perceptual skills. Parachutists must learn to accurately assess parachute opening status, sense visual motion cues, and to predict and manage their descent and drift toward the landing zone while avoiding obstacles. Traditional professional training techniques were limited to texts, lectures, procedures drills, and actual (but solo) flights. Physical skills can only be acquired through practice experiences, not classroom lectures.

Military aircrew receive maneuverable parachutes for emergencies, but any actual flight training provided is limited to a towed upwind low altitude launch under benign conditions at beginning of their careers. Subsequent training has been limited to lectures and procedures drills in a suspended harness, because emergencies are rare events and the hazards of real parachuting flights to a highly (and very expensively) trained individual. Essential parachuting skills and self-confidence could not be developed. The U.S. Air Force averages over 25 aircraft ejections per year and the U.S. Navy/Marine Corps over 50, of which over 90% are survivable. Of these, over half result in injuries or worse, and a significant number are related to parachute flight skills. Aircrew losses have enormous social, financial, political, and tactical costs.

Smokejumper facilities see the cost of injuries and fatalities to their civilian parachuting employees directly as a budgetary drain. A flight training simulator system based on low-cost PC-based technology for Computer-Aided-Design (CAD) was developed in collaboration with smokejumper instructors which met their training needs but stayed within their limited budget. They reported reduced training costs and improved parachuting performance and safety. This system has been deployed as a cost-effective training solution despite its relatively sparse scene detail capability.

Display Enhancements Requested

Feedback from military personnel being parachute simulator trained, while generally positive, had exposed some clear areas for improvements. Concerns were expressed with display limitations on control cue availability and mission training topics. With the older CAD-based graphics, severe compromises at the expensive of scene detail were made with the goal of maintaining frame rate, and thus simulated parachute responsiveness. This austere display technology, shown in Figure 1, provides tree pyramids, solid color building cubes, and most significantly, ground detail via grid lines. Research has shown that aircraft require ground surface microtexture cues for controllable low speed landings[8].

---

*Principal Specialist, Senior AIAA member ‡Vice President and General Manager of Systems Products Division
†President and Technical Director, AIAA Member
Copyright © 1997 by the American Institute of Aeronautics and Astronautics, Inc. All Rights reserved.

A more subjective concern is the overall look and feel provided in the scene display. In a real jump, a parachutist can look over a very wide range of horizontal and vertical directional angles. The computed scene motions had been presented to the trainee on a tilted monitor as a ground plane representation, with a severely limited field of view. While this method facilitated instructor-student dialog as shown in Figure 2 , it was felt that a more realistic replication in view scope and detail would in turn facilitate recall of vital skills in a critical moment.

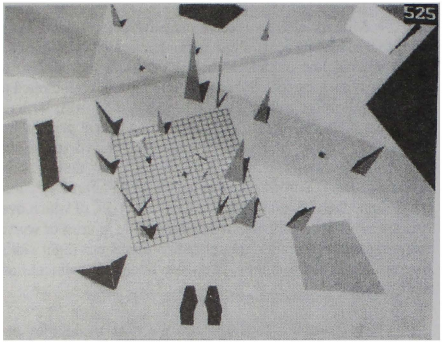


Figure 1. Typical Scene, Previous Graphics Technology

There is a generally held perception that the more realistic a training simulated appearance is provided, the more readily a trainee can make the transition from dealing with a purely abstract process to the view that real world skills are being practiced. An additional logical enhancement was a detailed and realistic representation of other aspects of parachute training such as malfunctions procedures, and the capability to present a simulation of actual expected mission terrain.

More Complete Virtual Reality Parachute Training Experience

In response to earlier requests for the addition of canopy collision training, and mission planning and rehearsal, a system was developed using expensive graphics workstations and dedicated simulation display generators[9]. While this met the functional requirements, the costs were more than an order of magnitude higher than the more austere system and seemed fiscally prohibitive.

Recent entertainment-driven trends and developments in 3-D graphics technology have lead to a steady migration of features and capabilities from dedicated simulation (primarily aircraft) display generators first to graphics workstations, and now to PC based 3D boards. These have now been adapted to parachute specific training needs.



Figure 2. Suspended Harness and Tilted Monitor

In particular, graphics coprocessor accelerator chip sets have been implemented in PC board products which provide the high-quality, interactive texture-mapping capabilities previously available only with dedicated graphics workstations and flight simulator display systems. The parachute flight simulator relies on a 3Dfx Interactive's Voodoo Graphics™ chipset which performs triangle setup and rasterization, while the system's host Intel Pentium CPU performs geometry and lighting calculations.

As installed on this manufacturer's Obsidian graphics boards, the simulator also takes advantage of the chipset's more advanced capabilities, including 16-bit depth buffering, comprehensive alpha blending with support for translucency and transparency at both the polygon and texture level, projected and detailed texture mapping, environment mapping, per pixel fog and associated atmospheric effects, texture animation, video texture mapping, fast linear frame buffer access, high-bandwidth texture paging, and support for 13 different texture formats, including 8-bit palletized and narrow channel compressed format.

Features	Benefits
Perspective-correct texture mapping	Reduces polygon counts mapping with Z-buffering and eases hidden-surface removal
Level-of-detail (LOD) MIP mapping	Eliminates texture aliasing
Bi-linear and advanced texture filtering	Eliminates pixelization
Texture compositing and morphing	Provides lifelike lighting effects and eliminates "object popping"
Animated texturing	Combines video with 3D
Anti-aliasing	Eliminates "stair-casing"
Polygonal-based Gouraud shading and texture modulation	Provides specular and diffuse lighting effects
Sub-pixel correction	Eliminates polygon and texture misalignment
Per-pixel alpha blending effects	Provides realistic atmospheric effects as well as translucency and transparency
Industry-standard OS and API	Simplifies software support development

Table 1. Specific hardware features and benefits of the 3Dfx graphics accelerator

Specific performance achieved includes:

- 45 Mpixels/sec sustained fill rate for bi-linear or advanced filtered textures
- 1,000,000 filtered, LOD MIP-mapped, Z-buffered, alpha-blended, fogged, textured 25-pixel triangles/sec on a P-166 PCI Intel based system
- 800 Megabytes/sec of graphics memory bandwidth

Depth complexity, a measure of the rendering complexity of a scene is the average number of times a pixel is rendered per frame. Typical depth complexity for current state-of-the-art hardware accelerated PC-based games is between 2 and 4. A game with a depth complexity of 3 at 640x480 at 30Hz requires fill rates of at least 27 Megapixels/sec. An entry level Obsidian board in this simulator system provides stable sustainable 30Hz frames rates with high depth complexity.

Even greater scene complexity at the same smooth high frame rates is possible. This chipset is now available implemented with multiple parallel texture processing units: dual Obsidian configurations with two to eight Mbyte texture memory and one to four Mbye frame buffer configurations achieve a 90 Mpixels/sec polygon/fill rate with 1.6 Gigabytes/second graphics memory bandwidth.

Complex mission specific scenes can be readily generated from digital map databases such United States Defense Mapping Agency Digital Terrain Elevation Data (DTED) and Digital Feature Analysis Data (DFAD), and photographic imagery. A variety of commercial tools are available for this purpose such as Gemini OpenGVS, Multigen GameGen, and Autodesk 3D Studio. The result is a simulation display scene with a very realistic appearance, minimal artifacts, and high frame rate.

As display generation system costs decreased by orders of magnitude, head tracking and display device costs have experienced a similar decline. Combining this with the existing simulation dynamics and control loaders produced a system, shown in use in Fig. 3, which provides an immersive subjective experience which has been referred to by its users as Virtual Reality (VR).

Several specific features of this technology enhance or add new parachute training capabilities:

Canopy Control Skills: The VR headset produces a compelling immersive environment which, combined with special ground texture effects, greatly facilitates the sensation of moving in a real 3-D world and provides better landing cues due to the strong perception of ground rush from the special layered surface microtexture as shown in Fig. 4.

Collision Avoidance: Most military parachuting operations involve two or more personnel. With a head mounted display and tracker, the trainee can keep track of and avoid turning into other parachutes, and other collision hazards such as trees, buildings, and power lines by scanning in all directions. Personnel can practice with their fellow team members using previously recorded runs or simultaneous network simulations.

Equipment Malfunction Procedures: An immediate and extremely urgent concern during parachute operations is evaluation of the canopy opening condition. Emergency procedures prescribe that personnel visually inspect the canopy, identify particular problem configurations, and follow specific procedures in a attempt to fix

them. Previously training was provided by viewing photos. With a head mounted display and tracker, the jumper can look direct up and observe the parachute canopy overhead as well as parachute's velocity and controllability. With the new graphics implementation, the parachute canopy displayed is composed of 3D polygon vertices. These are moved (animated) along paths to produce an object and its motions which resembles a specific canopy malfunction. The parachute dynamics model is manipulated to provide motions characteristic of malfunctions such as bag lock, premature asymmetric brake release, broken steering lines, rips and tears, line overs, etc.



Figure 3. Suspended Harness with VR Headset

The VR visor design allows a view of harness, etc., including the operational parachutist's main and reserve canopy deployment, and main canopy breakaway handles. Thus the new VR technology enabled the addition of a new simulator training capability which greatly extends its training scope and usefulness.

Mission Preparation and Rehearsal: Specific mission scenes can be created from digital map data: personnel can practice emergencies

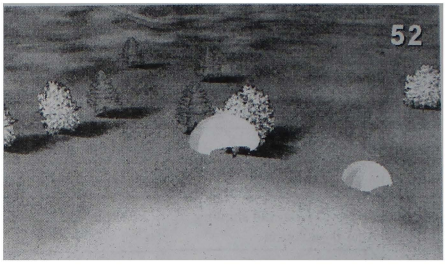


Figure 4. New Scene with Microtexture and Multiple Parachutes

resolution, wide field of view devices have also experienced significant sickness problems.

Sensory conflicts have been cited as a primary cause of simulator sickness. In a VR system, the display image must match head position, and any artifacts, delays or other dynamics could lead to serious conflicts between the visual image and head motions. The Fig. 5 block diagram illustrates how head tracker dynamics could produce a conflict with the visual image. In the current system the head tracker/filter dynamics are probably consistent with the graphics system time delay so that from the trainees point of view the visual scene does not appear to lag or lead the head motion. In any case, subjectively, the visual display/head motion mismatch seems minimal and the have been no reported instances of simulator sickness in hundred of exposures, probably due to a limited field of view a high 30 Hz update rate.

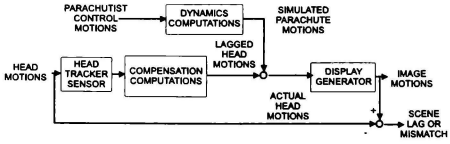


Figure 5. Head to Image Motion Mismatch with VR Head Mounted Tracker and Display

Fig. 6 shows how a flight helmet can be used in conjunction with a separated head mounted display and head tracker. In Fig. 3 they are combined with an adjustable headband.

but students are in fact taught to use risers to avoid collision with other paratroopers and terrain features. Operational jumpers shift their weight, much like a hang glider pilot, to use riser strap control to relieve the need to hold their arms up continually during long High Altitude High Opening (HAHO) flights, to counter malfunctioning standard controls, and to achieve additional trim and control effectiveness in special situations.

To provide simulator riser inputs, the trainee needs a 4-riser strap configuration suspended overhead. One way to do this, while improving a sense of face-validity, is to actually suspend the trainee in a 4-riser parachute harness as shown in Fig. 7.

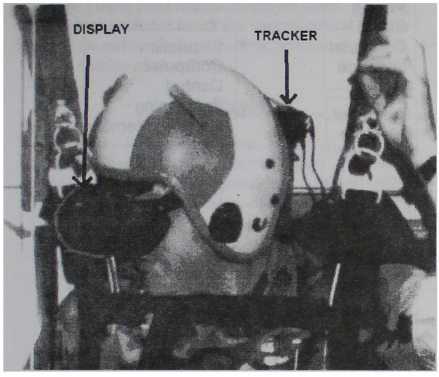


Figure 6. VR Display Worn under Actual Flight Helmet with Tracker Attached to Rear of Helmet.

The harness riser straps are attached via load cells to suspension lines. The student pulls on two control toggles lines routed to a force-control loader box. The control toggle signals, together with optional riser load and head tracker signals are routed to a high end PC-type computer as shown in Figure 8. The instructor can adjust fixed display simulation run orientation and post-run observation view point with a joystick.

The simulation computer systems contains high speed 3D circuitry, signal interfaces, disk data storage, and a simulation program which includes instructional and vehicle dynamics simulation features. The scene moves in appropriate simulated mathematical model response to the trainee's head motions, toggle control motions and variations in the differential riser strap loads by pulling on the straps or adjusting body position in the suspended harness. Main and reserve release and breakaway handle/cables can be sensed with appropriate simulation response. If no student malfunction response occurs, and an automatic opening device simulation has been selected, the system will simulate automatic canopy deployment. The simulated success or malfunction failure result is displayed as appropriate canopy simulated motion.

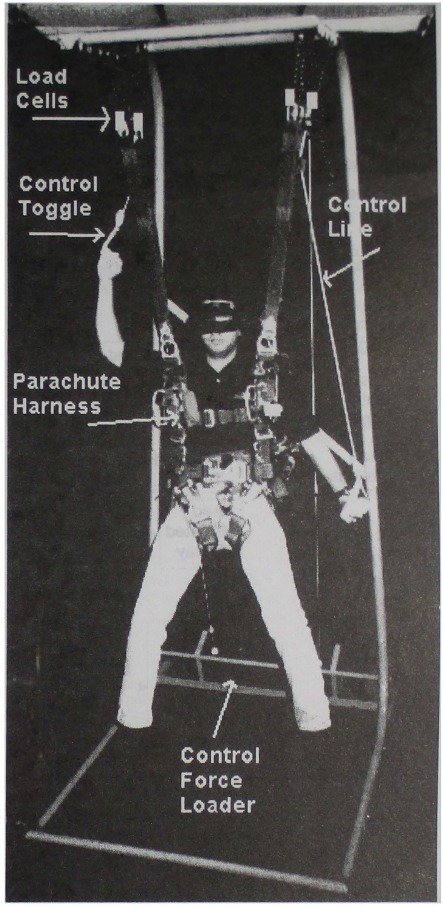


Figure 7. Suspended Harness with Load Cells on Riser Straps

A graphic display output is provided to the instructor for training control and monitoring, and a separate 3 dimensional full color graphics display scene output for the student. The instructional features [7] which were developed in collaboration with parachuting instructors, have been retained and enhanced. One example is the change from the ability to conduct joint training with previously recorded runs to true networked simultaneous operations with other trainees.

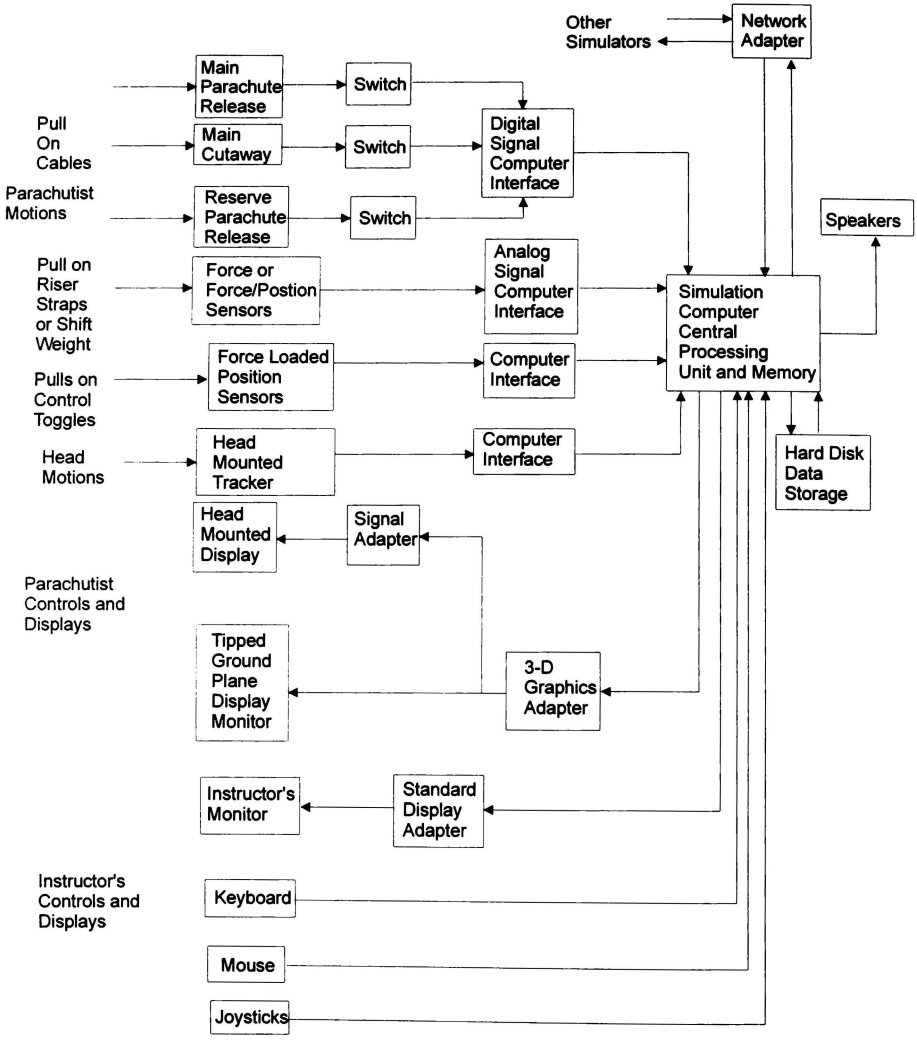


Figure 8. Complete Parachute Flight Simulator Block Diagram

Conclusions

It is both vital and practical to provide basic and refresher parachute flight training for operational parachutists and aircrew. A new Virtual Reality based system has been developed, installed and is being used for operational training. This was designed in specific response to solving concerns raised during current parachute simulator flight training. The new system provides a head mounted display and tracker combined with a much more realistic displayed scene. Training has been extended from basic canopy control skills to include malfunctions procedures, collision avoidance, and mission rehearsal.

References

1. Hogue, Jeffrey R., Johnson, Walter A., Allen, R. Wade, Pierce, Dave, *A Smokejumpers' Parachute Maneuvering Training Simulator*, AIAA-91-0829, American Institute of Aeronautics and Astronautics 11th Aerodynamic Decelerator Systems Technology Conference, San Diego, California, April 9-11, 1991

2. Hogue, Jeffrey R., Johnson, Walter A., Allen, R. Wade, *A Simulator Solution for the Parachute Canopy Control and Guidance Training Problem*, SAE Paper 920984, 1992 Society of Automotive Engineers Aerospace Atlantic Conference and Exposition, Dayton, Ohio, April 7-10, 1992

3. Hogue, Jeffrey R., Johnson, Walter A., Allen, R. Wade, *Parachute Canopy Control Simulation: A Solution For Aircrew Emergency Training* , STI Paper 473, Presented at the 29th Annual SAFE Symposium, Las Vegas, Nevada, November 11-13, 1991

4. *Parachute Simulator Reduces Risks of Injury on First Jump*, Aviation Week & Space Technology, October 26, 1992

5.*Dropping In On Aircrew Training*, Military Simulation & Training, March 1995

6. *Parachute Simulation*, Aviation Week & Space Technology, April 22, 1996

7. Hogue, Jeffrey R., Johnson, Walter A., Allen, R. Wade, Pierce, Dave, *Parachute Canopy Control and Guidance Training Requirements and Methodology*, AIAA-93-1255, RAes/AIAA 12th Aerodynamic Decelerator Systems Technology Conference, 10-13 May 1993, London UK

8. Hoh, Roger H.., *Investigation of Outside Visual Cues Required for Low Speed and Hover*, STI Technical No. 1213-1, 13 June 1984

9. *Computer-Parachute Mating Promises Pinpoint Accuracy During Jump Missions*, Armed Forces JOURNAL International, March 1993

10. Casali, John G., Wierwille, *"Potential Design Etiological Factors of Simulator Sickness and a Research Simulator Specification"*, Vehicular Simulation and User Behavioral Studies, Transportation Research Record No. 1059, 1986, Pages 66-74.

CAPABILITY FOR ANALYSIS OF CRUISE MISSILE SYSTEM COMPONENTS USING REALTIME SIMULATIONS

Craig P. Bouchard, Janet L. Broussard, Christina P. Gemmill, Richard D. Hartman,
Sara Moneyhun Leite, Eric Werme, David H. Wright
Naval Surface Warfare Center Dahlgren Division
Dahlgren, Virginia

ABSTRACT

A systems integration and prototyping capability has been developed that for the first time permits ship-launched cruise missile system-level laboratory testing and analysis in realistic, simulated dynamic environments. The capability is produced by surrounding tactical system components with realtime simulations of the sea, air, terrain, and ship and air vehicle avionics and dynamics. Critical modeling efforts involved the synchronized, multi-processor simulation of the simultaneous sensing of motion by the ship inertial navigation system and the flight vehicle inertial sensors to perform the transfer alignment. The required range of dynamic at-sea environments and hardware component errors can be simulated and used to drive different tactical system configurations to verify the system performance, including the results of the subsequent simulated missile flight to the target. The laboratory capability also demonstrates new concepts or proposed system improvements using prototypes or simulated system components. This capability was accredited by the governing program office Simulation Management Board. This involved extensive component-level evaluations against tactical data and reference simulation data, and system-level comparisons with data from at-sea tactical system testing. The realtime, simulated environment is shown to provide the realistic inputs needed to exercise the tactical system components over the range of dynamic conditions required.

1.0 INTRODUCTION

The successful integration of functional components to form a complex weapon system requires the application of the systems engineering discipline in all phases of the development and testing. Early demonstrations of system concepts using tactically-representative environments provides insight and the opportunity for concept refinement to the designers of the system and the eventual users in the fleet. Once the development progresses to having the system hardware and software implementation in hand, a realistic test and analysis environment is required to verify, as early as possible, the compatibility and integrated performance of the tactical system components before proceeding to more expensive system-level testing involving operational platforms. For the case of a sea-launched cruise missile system, the at-sea testing can involve the scheduling and use of ships, planes, specially-instrumented air vehicles, and test ranges. However, the computer hardware and software, interfaces, sensors, and some communication components of the system may be substantially evaluated in the computer laboratory, provided that realistic inputs are made to the sensors or sensor interfaces. This paper presents a new laboratory capability developed to provide a high-fidelity simulated environment to exercise the shipboard and missile guidance components of a sea-launched cruise missile system.[1] While the primary focus of this capability is on analysis to assess systems performance and complement more expensive at-sea flight tests, prototyping of new or advanced systems concepts is also done in this facility to demonstrate system compatibility or predict performance.

The approach taken in developing this capability uses the synchronized, realtime execution of simulations of the ship navigation systems and dynamics, missile sensors, airframe, and the flight environment. This allows in-laboratory operation of essential tactical system components over a large range of simulated environments and scenarios. While a perfect realization of the environment is not possible in simulation, the simulated environment offers the advantages of determinism in scenario conditions, expanded insight into system operation, and the potential to use large test sets to ease the pressure on operational test programs to cover the envelope of required operating conditions. Fulfilling the goal of complementing limited at-sea test programs and verifying performance early in the development process requires operating the weapon system components in a simulated environment with sufficient fidelity to excite the system responses possible in actual environments. For the system components exercised in this laboratory, the dynamics of the pre-launch transfer alignment process define the fidelity requirements for the shipboard environment simulation, and the flight navigation and control dynamics set the requirements for the simulated flight environment. Comparison of the results from the hybrid tactical/simulation system with data from previous "live" tactical system tests

This paper is declared a work of the U.S. Government and is not subject to copyright protection in the United States.

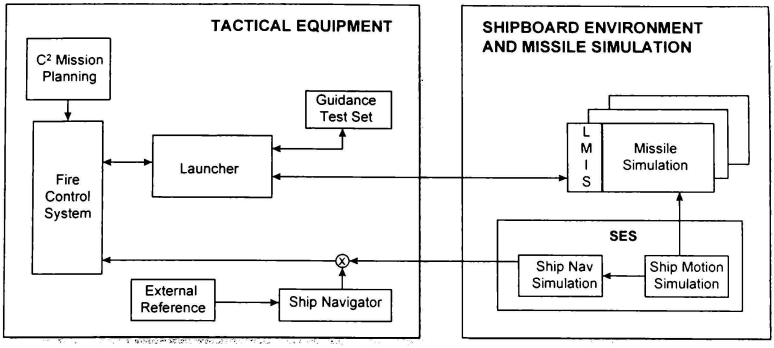


Figure 1. Cruise Missile System Integrated Test Laboratory

provides confidence that the system performance in various operating conditions can be accurately characterized and quantified in the laboratory.

For the purposes of this paper, the cruise missile system discussed is functionally grouped into the three component areas of Command and Control/Mission Planning (C^2), Fire Control/Launcher, and the Missile. The C^2 functions include the attack planning, air vehicle route planning, and mission evaluation activities that produce the appropriate data for the missiles to use during the subsequent flight. The Fire Control/Launcher system (FCS) functions prepare the missile for and initiate the flight; including generating quick response route plans, loading computer programs and data to be used during flight, and assisting in alignment of the missile inertial navigation systems using the shipboard navigation system. The missile performs the preparations for flight in conjunction with the FCS and then accomplishes the flight to the target using onboard guidance and the mission planning data.

This paper is organized as follows: the laboratory concept is presented in Section 2, the modeling challenges and the details of the modeling implementation are described in Section 3, the process of accreditation and presentation of the simulation performance are addressed in Section 4, and conclusions on the performance of the developed capability are presented in Section 5.

2.0 LABORATORY CONCEPT AND ARCHITECTURE

The developed laboratory capability addresses the need to exercise the weapon system and interfaces in the laboratory in the same manner as when they are deployed at sea. This is accomplished by blending the cruise missile tactical hardware system with various simulations to perform integrated system-level operations. Figure 1 gives a block diagram representing the components of this laboratory configured for ship-launched cruise missile systems integrated testing. The laboratory consists of the communications systems employed between ashore systems and afloat systems and among the afloat system, and the complete tactical hardware for the C^2 and FCS components of the system connected by realtime interfaces to a high-fidelity simulation of the shipboard environment and the missile. To exercise the tactical components in realistic scenarios, the interfaces to components not included in the tactical suite (e.g., to sensors such as accelerometers, receivers or antennas) are provided with data that is consistent with the tactical environment of the scenarios. Thus, the environment for these components includes simulations of ship and missile navigation system sensors, missile control sensors and other avionics that have interfaces with the guidance computer, and the dynamics models needed to provide information to the sensor simulations.[2] The actual gyros and accelerometers are not used, eliminating the requirement for motion tables to drive those elements. Instead, dynamics information must be provided at the high data rates and unique specifications of the sensor interfaces.

The Shipboard Environment and Missile Simulation (SEMS) includes the simulation components of the laboratory, and consists of a Ship Environment Simulation (SES) and a Missile

Simulation (MS). The MS provides a complete representation of the missile so that the FCS operates as if connected to a real missile, via the launcher, and to the ship navigator. It includes a register-level emulation of the on-board missile guidance computer, and provides a complete six degree-of-freedom (6 DOF) simulation of the missile and all environmental conditions affecting its performance. The SES provides the same communications to the FCS that the shipboard navigation system does, representing ship performance under various conditions. Both simulations operate in realtime and maintain realtime interfaces with the tactical hardware. The ship motion data generated by the SES is communicated in realtime to the MS so that the simulated missile inertial measuring unit (IMU) components (i.e., accelerometers and gyros) sense the same ship motion as reported to the FCS. The SES can be configured to simulate the motion of any ship class that is equipped with the weapon system, positioned anywhere in the world, in a variety of sea states. The tactical connection of the FCS to the missile is provided by the launcher via a missile umbilical cable. In the SEMS, the Launcher-to-Missile Interface Set (LMIS) simulates the missile canister functions and provides the physical connection between the launcher umbilical and the Missile Simulation.

Due to the maturity of the weapon system at the time of the development of the SEMS, there were existing hardware and software components which met many of the requirements for the system. Simulation reuse is a major goal of the SEMS development and has resulted in the integration of several "reused" simulations, providing reduced cost, development time, and accreditation effort, as well as increased reliability and maintainability. A non-realtime simulation, previously accredited by the program office Simulation Management Board, was selected as the basis for the Missile Simulation. A ship navigation system simulator used for on-board ship FCS testing was the basis for the SES. Hardware boards from another missile simulator were the basis for the LMIS. In each case, the implementation for the SEMS required some modifications to each of the reused components as well as the development of new capabilities. The only completely new component is the SEMS Controller. This program directs the execution of all of the components of the SEMS, performs remote data collection and support functions, and provides a single point for operator control of the system. A wrap-around simulation program (WASP) was developed as part of the SEMS Controller. The WASP allows the SEMS to be executed without requiring any of the tactical shipboard equipment. For a development that includes tactical components, a WASP is required due to the limited availability of the tactical systems and the resources expended to perform realtime man-in-the-loop tests.

Simulation reuse greatly influenced the hardware design of the SEMS (shown in Figure 2). The equipment reused for the LMIS consisted of three custom VME interface cards. The VME architecture offered considerable flexibility for development, so the existing MS program was ported to run on a VME single-board computer installed in the same chassis as the LMIS. Since the ship navigation simulation used for the SES executes on an IBM-compatible PC, a shared memory network was configured to facilitate the SES to MS interface. This led to a straightforward design requiring minimal changes to the existing simulations. Use of the shared memory network also facilitated the implementation of the SEMS Controller hosted on a TAC workstation, and provided connectivity with the special-purpose workstation used for the realtime visualization system.

The SEMS processing is initiated in concert with the FCS operations involving the missile and shipboard navigation interfaces. At missile power-up, the missile guidance computer executes a bootstrap loader program from ROM that facilitates the downloading of the complete Operational Flight Program (OFP) from the FCS, then transfers control to the OFP after the download. To precisely model the behavior of the missile, the MS emulates in realtime the execution of each machine instruction in the bootstrap loader program and in the downloaded OFP. At the same time, the simulation of the missile IMU is stimulated by realtime ship motion data from the SES. The pre-launch operation of the SEMS includes alignment of the missile IMU, download of the mission data, and preparation for launch. Once the FCS initiates the launch, the realtime launch process is simulated by the SEMS. At the moment when the missile exits the launcher it is no longer affected by the launch platform (e.g. ship), and the MS interfaces to the FCS and the SES are broken. The SEMS non-realtime flight simulation is initiated from a checkpoint saved at launcher escape, and can be executed at a later time or repeatedly under varying simulated flight environmental conditions. The flight simulation includes a full 6 DOF representation of the missile and its environment as controlled by the OFP through the transition from boost to cruise flight, the autonomous navigation and route steering, and to the flight termination. Throughout the realtime pre-launch and the non-realtime flight phases, the SEMS MS captures data from both the OFP and the environment model for analysis and visualization. The simulation is terminated at the end of the missile flight.

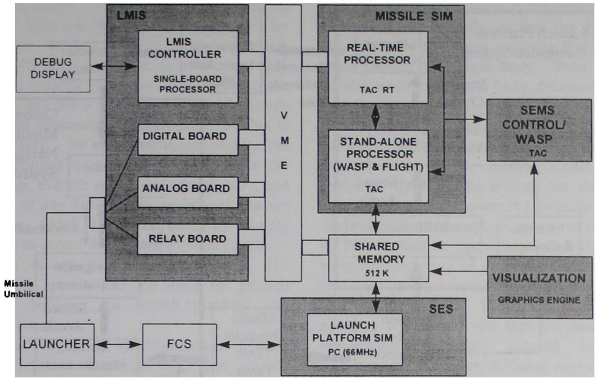


Figure 2. SEMS Architecture

3.0 MODELING CHALLENGES

The primary technical challenges faced in hybrid simulation system operation (tactical system components with simulated components) are to synchronously execute multiple, high-fidelity simulations and simultaneously support the requirements of the tactical interfaces. The different approaches available to address these challenges are, in this case, constrained by the goal to adapt existing, accredited simulations for this application to the extent possible. This section discusses the simulation challenges faced in this development and the chosen implementations.

The modeling challenges encountered in this development can be grouped into three areas. One area is maintaining communication between the missile simulation and the FCS in accordance with the appropriate interface specifications. Specifically in the MS, the execution of the OFP using the guidance computer emulation must be controlled so that for every execution cycle of real time the correct number of machine instructions are executed and the simulated clock used by the OFP is adjusted by exactly the time of the execution cycle. Data flows through the serial data interface between the FCS and the MS at a fast rate and the OFP must process the incoming data. So, whenever data is received by the MS from the FCS, the OFP must be allowed to execute immediately so that a response can be returned to the FCS within the interface time-out period. The major challenge in this area is assuring that the MS accurately represents the low-level timing functions of the guidance computer and satisfies all realtime interface requirements.

Another modeling challenge involves the coordinated modeling of the true launch platform and missile dynamics during the pre-launch phase. This modeling area, together with the challenges of maintaining the realtime interfaces, determine the accuracy of the representation of the critical transfer alignment process performed jointly by the FCS and the missile. The discussion of the significant SEMS pre-launch modeling relative to this alignment process is presented in Section 3.1.

The final challenge faced in this development is modeling the transition from the pre-launch dynamics, driven by the launch platform motion, to the free-flight missile dynamics while satisfying the launch platform analog checks for the simulated missile separation from the launch platform. The modeling of this transition and the flight from launch to the target are discussed in Section 3.2.

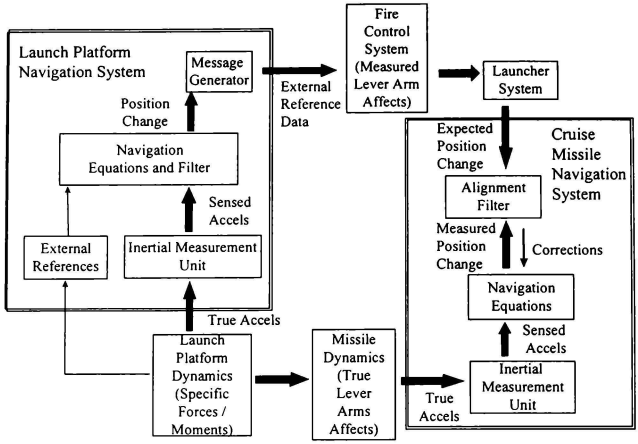


Figure 3. Transfer Alignment Loop

3.1 MODELING OF THE TRANSFER ALIGNMENT

Before discussing the modeling support for the transfer alignment process, a brief description of that process is useful. The alignment of the missile inertial navigation system in a dynamic environment requires an external reference source of equal or better navigation accuracy than the system being aligned. In the case of a cruise missile launch from sea-based surface and subsurface launch platforms, the process of transfer alignment involves comparison of position change data from two navigation sensors at different locations on the launch platform. The data from the launch platform navigation system, the external reference, is provided to the cruise missile via the FCS. The FCS primarily serves as a conduit to the missile, compensating for lever arm effects to compute the expected position change at the missile inertial sensor location. The missile sensed acceleration data is integrated twice to determine the missile sensed position change, which is compared with the expected position change provided by the FCS. The residual between the expected and computed position changes is used by the missile alignment filter to refine the calibration of the missile navigation system[3]. As the filter refines its estimates of the navigation system errors, the difference between the expected and computed position changes tends towards zero indicating that the missile system is ready to begin autonomous navigation. This process is shown in Figure 3.

In order to simulate the transfer alignment, it is necessary to simulate the dynamic environment, the characteristics of the master navigation system, and the characteristics of the missile navigation system. The level of fidelity and the appropriate motion in the simulations are determined by the sensitivity of the alignment filter. Furthermore, the modeling assumptions and execution of the launch platform and the missile simulations must be matched so that modeling differences and asynchronous effects are not perceived by the alignment filter as navigation system errors. Small errors due to synchronization become significant when there are high dynamic rates and when the synchronization must be maintained for long periods of time.

The key modeling decisions affecting the alignment process are the distribution of the launch platform and missile truth models between the processors and the synchronization of the processors. Two options are apparent. The first is to have all of the truth computations in one of the processors. The

benefit of this is that multi-processor synchronization is not required. The drawback is that additional weapon models added to the architecture must be incorporated into the existing model. This limits the flexibility of the system and increases the implementation cost for additional missile simulations. The second approach, that is the SEMS implementation, is to execute each truth model in a separate processor. The model of the launch platform and the 6 DOF missile model execute on two different processors. The benefit of this is increased flexibility. By executing independently, the models are initialized at the appropriate times in the system operation, execute at the appropriate rates, and can be stopped at any time. Also, additional weapon models can be added to the system by simply modifying them to accept the initialization and dynamic data from the launch platform simulation. The drawback of this approach is the need for explicit synchronization between the launch platform and missile simulations.

The required degree of synchronization between the launch platform simulation and the missile simulation is determined by the total amount of time to be simulated and the severity of the dynamic environment. The level of agreement between the two simulations is a function of the numerical integration scheme employed in each, the execution rate of each, and the data senescence in the interface between the two. The greater the degree of agreement between the two simulations in these characteristics, the less synchronization is an issue. For this application, the interface between the simulations is via the multi-processor shared memory network, which provides flexibility in implementing the coordination of the dynamics modeling and induces senescence on the order of just nanoseconds. This is negligible compared with the other error sources and is not discussed further. The method of synchronization of the simulations is achieved either through driving the processes from a common computer interrupt or by assuming that the launch platform model is the "true" solution and slaving the missile dynamic solution to the launch platform motion using parameters in the shared-memory. Using a common realtime interrupt for both the launch platform and the missile simulation guarantees the synchronization of the two simulations. However, the required coordination of the simulations execution rates is too constraining for this application. The current missile execution rate is significantly lower than that anticipated for future additions to the laboratory. Those additions include missiles using strapdown inertial navigation systems[4] that operate on different time bases and at rates of several times the currently implemented missile processing rate.

The selected approach for the SEMS assumes the launch platform model is the "true" solution and slaves the missile dynamic solution to the launch platform motion. The missile, when in the launcher, is assumed to be mounted securely, so that the dynamics of the missile can be computed assuming that the missile is a point located on a rigid body. Prior to launch, the missile model uses rigid body dynamics and the lever arm between the launch platform navigator and the missile inertial sensing point to determine the missile navigator dynamics based on the launch platform dynamic data. The missile model modifies the computed dynamics to account for missile hardware errors and transmits measured velocity information to the OFP. The first implementation of this approach in the SEMS uses an open loop in which the numerical integration schemes of the simulations are identical so that only timing inaccuracies cause the simulations to lose synchronization. The launch platform simulation executes at a higher rate than the missile simulation minimizing the error induced by the timing inaccuracies. However, different execution rates mean different time steps in the numerical integration schemes. If the launch platform accelerates at a constant rate, the acceleration will be integrated over a slightly different time in the launch platform simulation than in the missile simulation due to the different time steps. The result, after integrating twice in both simulations, is a change in the relative position of the missile with respect to the ship. This is a modeling error given the rigid body dynamics. For short execution periods, these errors tend to remain small.

For longer execution periods, the increase in the position difference between the two dynamics models in the open loop approach begins to affect the alignment performance. With the open loop synchronization, the only solution is to tighten the requirements on the agreement between the integration scheme and the time step. This may severely limit the flexibility of the system since most high-fidelity simulations have assumptions that depend on either the time step or the integration technique or both. Another method of slaving the missile dynamics to the launch platform motion uses the rigid body dynamic relationships in order to form a closed loop system that constantly makes small dynamics adjustments to account for computer modeling drift. The benefit of this approach is that it should maintain synchronization indefinitely and does not require any adjustments to the simulation time step or integration scheme. The drawback is that the missile simulation model has to be further modified to incorporate the correction terms. This approach is being developed to support longer execution times.

The SES models the launch platform dynamic motion and various launch platform navigators at different levels of fidelity. The first order model includes a limited set of launch platform navigator error parameters and sea motion effects, while the more detailed model includes master navigator external updates and an expanded set of navigator error parameters and sea motion effects. This higher order model is required when a detailed performance analysis of tactical components integral to the alignment loop is being conducted. The first order model is the appropriate level to use when evaluating components of the tactical system outside of the alignment loop and is the modeling represented in results given in this paper. With this model, the launch platform accelerations and angular rates, computed at the location of the launch platform navigator, are analytically derived from the kinematic relationships contained in the reused launch platform simulator. The launch platform navigator model simulates the sensing of inertial accelerations and attitude and produces the sensed or measured data needed by the FCS. The navigator model also allows the application of operator input errors and biases to the true dynamic data to produce the measured data.

3.2 MODELING OF THE FLIGHT FROM LAUNCH TO TARGET

At the beginning of the Launch to Target phase, the missile transitions from being held within the launch platform to being a free rigid body. Following booster ignition, while the missile is still in the launcher, motion by the launch platform still influences the dynamics of the missile. However, the missile and the launch platform begin to act as two separate rigid bodies in contact with one another. During this period, the event timing for the analog signals indicating the missile separation from the launcher must be satisfied. This requires modeling the restraining forces applied to the missile after the booster has ignited and prior to separation. As the missile tail clears the launcher, the SES and the MS are decoupled.

Once launch separation has occurred, the missile 6 DOF simulation executes the guidance flight software using the information loaded into simulated guidance computer memory during the pre-launch phase and simulates the surrounding environment and missile hardware until flight termination. Control surface response to autopilot commands are simulated using a first order response model of fin angular accelerations. The aerodynamic forces and moments, propulsive forces and moments, and mass loss are determined using functionalized 6 DOF databases. Missile mass and inertia properties are continuously adjusted to account for fuel expenditure, and changes in the physical configuration of the missile airframe. The missile earth relative position and orientation are maintained relative to a rotating WGS-84 oblate spheroid earth model, as is the gravity field. Air vehicle sensor models are driven by the resulting dynamics solutions, the atmosphere and winds model, the Digital Terrain Elevation Data, the models of the external navigation references, and others.

4.0 ACCREDITATION

Before the simulation could be used in tactical system testing, accreditation by the governing program office Simulation Management Board[5] was required. Supporting this board are technical review panels tasked to evaluate the various simulations and recommend accreditation if warranted. To achieve accreditation, the simulation developer must create and have approved an accreditation plan, a configuration management plan, and a simulation description for inclusion in a simulation catalog. The results from tests outlined in the accreditation plan are then presented to the technical review panels for evaluation. Upon acceptance by the appropriate panel, accreditation is recommended to the Simulation Management Board which grants certificates indicating the method of accreditation, the purpose of the simulation, and any limitations of its application. Due to the complexity of the SEMS, accreditation was achieved by reviewing the results from component tests and from total system level tests[6].

For accreditation of the SES with first-order dynamics modeling, interface tests and dynamic motion modeling tests were performed. Data was collected from the SES to FCS interface to verify that operator inputs and requested maneuvers were accurately reflected in the data and that the format and periodicity of the data messages sent to the FCS met interface specifications. To verify the dynamic motion modeling, the outputs of the model were compared against reference simulation data and data collected during actual at-sea tests. This data was also reviewed by the independent developers of the high-fidelity reference ship-motion model. Data was compared for different ship classes, sea states, speeds, headings, positions, and maneuvers. The data was not expected to match exactly due to the first-order launch platform motion modeling and the difficulty in knowing and matching the exact conditions of the at-sea tests. Comparisons validated the behavior and characteristics of the significant navigation parameters under the specified conditions.

Since the MS was based on a previously accredited simulation, the SEMS accreditation testing only involved the capabilities that were added. Timing tests,

software and mission data load tests, interface tests, launch performance tests, and boost performance tests were performed to validate the changes to the MS. To verify realtime execution of the OFP, timing data was collected and compared against the same data collected from an actual guidance set. Software and mission data loading were verified by comparing the contents of the guidance memory after software and mission data loading with the same memory contents of the accredited simulation containing the same software and mission data. Data was collected from the launcher to missile interface and from the SES to MS interface to verify that the transmitted data was correct and that all interface timing and protocol requirements were met. Launch performance testing involved collecting data from the launcher to missile interface during the launching process to verify that discrete signals to and from the launch system occurred as specified in the respective launcher and OFP documentation. Data was also collected from the MS after transition to flight and compared with data collected from the accredited, community-wide reference (non-realtime) simulation to verify that the MS was properly initialized for flight and that the transition from realtime to non-realtime processing was successful. Timing data was collected to verify that the realtime performance was not affected by the launch sequence. Enhancements to the MS boost phase modeling were verified by comparing boost data collected from the simulation with reference boost data.

The overall SEMS configuration was accredited by performing system integration tests, alignment tests and "end-to-end" (i.e., from mission preparation to flight termination) tests. The purpose of the system integration tests was to verify that the FCS could operate with the SEMS as with an actual ship navigator and missile (i.e., successfully launch the simulated missile under various tactical configurations and respond as expected to faults induced in the simulations). The alignment tests involved collecting data from the launch platform and missile models and comparing time histories of significant navigation and alignment parameters with data collected from alignments of a real guidance set in the laboratory and from ships at sea. During the alignment tests, various ship classes were modeled, and alignments were performed under maneuvering conditions representative of the actual at-sea alignments. In order to compare the simulation data with data collected from the missile guidance set, the simulated missile IMU hardware error model needed to be set to match the estimated hardware errors and characteristics of the real IMU. There are many interrelated IMU model error parameters that may be modified to accurately represent an actual missile guidance set; however, the problem of adjusting these parameters is a nonlinear optimal matching problem. The optimization problem was not addressed for this accreditation. Instead, three of the most signification hardware parameters were tuned to represent the behavior of a static alignment of the particular guidance set used to generated the reference data. The effects of tuning one of the hardware parameters is illustrated in Figure 4. In this figure, the alignment filter estimates of the hardware parameter (gyro drift) for the tuned and untuned IMU error model are compared with the results from a static alignment of the actual reference guidance set. The simulation was only tuned to produce a match of the dominant alignment characteristics. Tuning of additional, second-order IMU error model parameters is needed to produce a match in the filter-start-up response. Despite the limited tuning of the IMU hardware model, the important alignment variables collected during the static alignments, such as the east component of the filter update residual seen in Figure 5, compare well with the reference data.

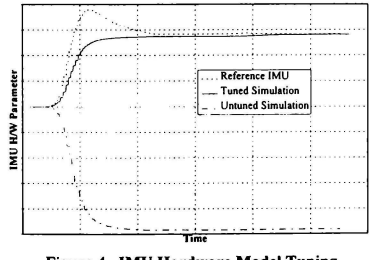


Figure 4. IMU Hardware Model Tuning

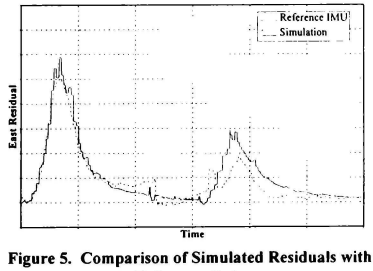


Figure 5. Comparison of Simulated Residuals with Reference Data

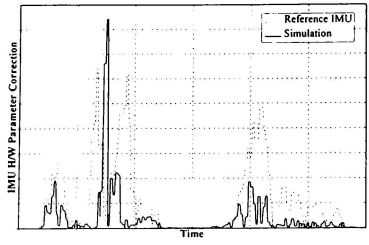


Figure 6. Comparison of Simulated Hardware Compensation Parameter with Reference Data

To further ensure that the SEMS could duplicate the characteristics of the at-sea alignments, comparisons were performed to validate general behavior and characteristics of the significant alignment parameters under dynamic ship motion conditions. Prior to the SEMS tests, the motion of the ship during the at-sea tests had to be estimated from the existing data collected from the ship navigator during the tests. Then, during the SEMS tests, the simulated ship had to be manually driven to match the recorded ship heading, speed, maneuvers, etc. as closely as possible. Because of the first-order modeling and since the exact conditions at sea could not be known, the simulation data was not expected to match the reference data exactly. For example, during one of the reference at-sea alignments in which a 90 degree turn was performed, the ship course and heading diverged after the turn due to the effects of ocean currents. Because the SEMS did not model ocean currents in the first-order launch platform simulation, the position changes computed by the SEMS did not always closely match the reference data. Despite this, the alignment parameters examined for the comparison with at-sea data did exhibit the same general behavior as the reference data as seen in Figure 6.

The purpose of the end-to-end accreditation tests was to verify that the simulated missile could be prepared for launch by the tactical FCS, and accurately represent the subsequent launch and flight to the target. To accredit the SEMS in this regard, flight data was collected from the MS and compared with data from the reference missile flight simulation. In this case, knowing the exact environmental conditions that produced the reference simulation data allowed a more specific comparison with the SEMS results. To demonstrate this comparison, Figures 7 and 8 show some commonly compared flight parameters. Figure 7 presents the boosted flight phase missile altitude and body axial acceleration profiles with respect to time. The acceleration profiles match closely, with slight differences arising in the altitude profile during the transition to cruise flight. Figure 8 presents the clearance altitude and normal acceleration of the missile during a period of terrain following. Because slight differences in missile speed between the two data sets, over a long flight, change the time that the terrain variations are encountered, the data was first synchronized at a ground point corresponding to the center of this plot. The comparison between the SEMS clearance altitude and normal acceleration and those from the reference simulation is quite close, with separation developing due to the differences in the terrain being overflown. The good agreement between the SEMS and reference data at this point of the flight also provides a verification of the performance of the SEMS architecture and modeling in the earlier pre-

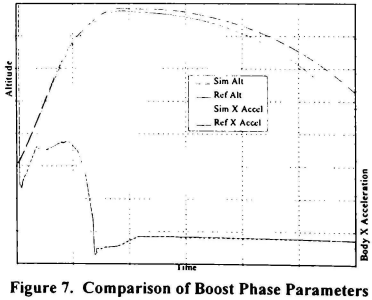


Figure 7. Comparison of Boost Phase Parameters with Reference Data

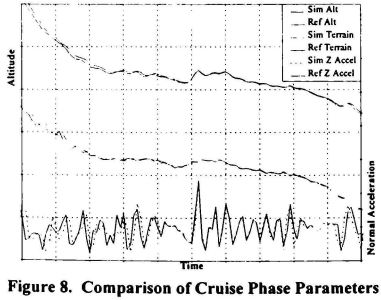


Figure 8. Comparison of Cruise Phase Parameters with Reference Data

launch and transition-to-cruise phases of the tactical system operation.

5.0 CONCLUSIONS

A new laboratory systems analysis, integration and prototyping capability has been developed by surrounding tactical cruise missile system components with simulations of the sea, air, terrain, and ship and air vehicle avionics and dynamics. Existing, accredited simulation programs are reused to a large extent. The capability to assess the performance of the transfer alignment process has been achieved using coordinated, multi-processor simulation of the simultaneous sensing of the ship motion by the launch platform inertial navigation system and the flight vehicle inertial sensors. This capability provides for the integration and analysis of tactical system components early in the development process, and complements more expensive and schedule-limited at-sea testing. The laboratory simulation system is accredited by the governing program office Simulation Management Board, involving technical community review and evaluation of the hybrid systems performance against data from standalone tactical components, reference simulations, and at-sea testing. This realtime, simulated environment provides the realistic inputs needed to exercise the tactical system components over the range of dynamic conditions required and to assess the system performance in laboratory end-to-end tests.

6.0 REFERENCES

1. *Strike Warfare Systems Integration Laboratory Concept of Operations*, NSWCDD/TR-94/153.
2. *Strike Warfare Systems Integration Laboratory Shipboard Environment and Missile Simulation Segment Hardware/Software Requirements Specification*, NSWCDD Report T-BSN-0457-00, August 1994.
3. Geyer,E.M., Miller,J.H., "Kalman Filter Functional Analysis (Preliminary Report)", TASC TIM No. 4463-1, February 1983.
4. Savage,P., "Strapdown Inertial Navigation Lecture Notes", February 1990.
5. *TOMAHAWK Simulation Management Plan*, PEO(CU)5232/3, June 1994.
6. *Shipboard Environment and Missile Simulation (SEMS) Accreditation Report*, NSWCDD Report T-BSN-0466-01, September 1996.

LESSONS LEARNED IN THE USE OF REAL-TIME SIMULATION FOR TECHNOLOGY ASSESSMENT

Brian K. Stadler* and Joseph P. Nalepka†

Wright Laboratory
Wright-Patterson Air Force Base, Ohio 45433

ABSTRACT

Due to the increasing cost of technology development, both virtual and constructive simulations are used to make mission effectiveness assessments of various technology attributes. These types of studies usually have been done by specialized groups using special software packages and simulation facilities. However, as computer computational and graphics power has increased, more individuals and groups can now execute software using a PC or workstation that once took specialized and expensive computers as well as specially trained engineers. Unlike constructive simulations, virtual operator-in-the-loop mission effectiveness or technology trade studies require far more attention than one might believe. Problems involving real-time software execution, limited field-of-view displays, model fidelity, and test biases are just a few of the issues one must be concerned about when developing multi-player combat simulations. Wright Laboratory recently conducted a one-versus-one study investigating the synergy between an agile aircraft incorporating thrust vectoring, helmet mounted displays, and high off-boresight agile missiles. Many of the lessons learned and problems encountered are typical of larger multi-player combat studies and provide insight into issues engineers should be concerned about when developing and conducting these types of studies.

* Aerospace Engineer, Member AIAA
† Aerospace Engineer, Member AIAA
This paper is declared a work of the U.S. Government and is not subject to copyright protection in the United States.

INTRODUCTION

There is little information in the literature on the problems and lessons learned in using piloted simulation for conducting mission effectiveness or technology trade studies. The experience at Wright Laboratory has been that these studies are extremely difficult, require far more diligence and planning, have far more tradeoffs, and require close attention to design of experiment practices. Combat simulations usually involve line pilots who may not be as forgiving of simulator nuances, especially when it effects their combat performance. Flight simulation and simulation in general is now being used more and more to determine future investments, make decisions, and determine which technologies have the most payoff. Bad simulations may lead to bad decisions, or worse, disbelief in simulation results and simulations in general

The use of workstations, and in some cases PC's, to conduct man-in-the-loop (MIL) simulations for mission effectiveness and technology trade studies is becoming more prevalent. This is due to the continually increasing computational and graphics power of workstations and PC's, as well as the ease at which they can be programmed and linked together. There are a few drawbacks to workstation and PC simulations in that they are generally non-real-time, have non-deterministic software frame rates, and may rely on extremely limited field-of-view visual scenes. These

drawbacks make them unsuitable for some types of air combat. However, these drawbacks have not limited their use and acceptance as tools for conducting simulation studies. In contrast, many existing government and industry real-time MIL simulation facilities were constructed to support aircraft flight control system development and flying qualities research. Within the past decade, even these facilities have integrated their separate simulators together in order to conduct studies generally accomplished via MIL-constructive or constructive workstation simulations.

A few of the MIL workstation combat simulations are modified versions of constructive air combat packages such as TAC-Brawler and AASPEM (Air-to-Air System Performance Evaluation Model). These constructive simulations are excellent tools for investigating the mission effectiveness of different technology attributes and performing numerous trial runs for generating statistical databases. Because this process can be very time consuming, the ability of these packages to execute without much oversight is a benefit. The drawback to constructive packages is that the pilot decision logic or mental model may impact how a technology is used and requires some programmer expertise[1]. In addition, modifying the constructive packages with new models to improve the fidelity or add new capabilities can be difficult. The addition of a MIL capability to many of these packages is an attempt to bring human variability into the tests or to use the constructive package as a inexpensive alternative to piloted dome simulations[2].

The Wright Laboratory Flight Dynamics Directorate simulation facility, like many others, was constructed around specialized analog and digital real-time computers and graphics processors. These computer systems were required to achieve frame rates of 40Hz or higher and simulation latencies of 100ms or less[3]. These strict requirements were necessary from a modeling, test, and evaluation standpoint, since the poor modeling of the aircraft or its flight environment can adversely affect the pilot's opinion of an aircraft's flying qualities and his acceptance of the accuracy of the simulation. Conversely, since flying qualities simulations tend to use test pilots, their concern over cockpit realism is much less than aircraft dynamic accuracy. The computers and image generators used in the dome engineering facilities have also increased in computational power thereby increasing their capability and flexibility. Aircraft models which used to require two or three CPU's to execute now only require one CPU with computational capacity still remaining on that CPU for other functions. This increase in computer performance has allowed many of the engineering facilities to perform studies similar to the constructive simulations or be used to augment those studies[4]. This paper will discuss some of the problems and lessons learned during the development and test of a simple one-versus-one combat study.

EXPERIMENT DESCRIPTION

Due to the recent success of several flight demonstration programs, Wright Laboratory was interested in investigating the potential synergy between an agile aircraft with thrust vectoring (TV) capability, helmet mounted displays (HMD), and high off-boresight agile missiles (HOBM).

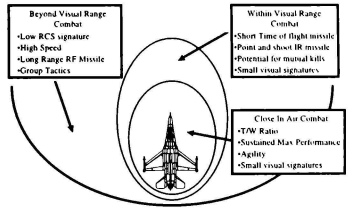


Figure 1 Air Combat Regions

These technologies have matured to such a point that the potential exists for retrofitting these technologies to existing aircraft and dramatically increasing their within visual range combat performance. The Multi-Axis Thrust Vectoring (MATV) Program demonstrated the integration and utility of a near production axisymmetric thrust vectoring nozzle using the Wright Laboratory VISTA/F-16 aircraft[5]. Similarly, using a different F-16, the "Look and Shoot" program integrated a Honeywell HMD and magnetic tracker with a Raytheon high off-boresight missile for live fire demonstrations[6]. Unfortunately, neither of these programs used the same F-16 and there was interest in how a combination of these technologies would work during within visual range combat.

Generally, air combat is broken into three regions: Beyond Visual Range (BVR), Within Visual Range (WVR), and Close In Air Combat (CIAC). Care must be taken when developing combat simulation tests because each of these regions has its own characteristics which will drive the requirements of the simulation and the test setup (Figure 1)[7]. Technology characteristics such as short time of flight missiles or highly agile aircraft may dominate and unrealistically overshadow other technologies. In addition, simulation artifacts and biases become more pronounced and may become critical interactions that require consideration during experiment design.

Due to these reasons a one-versus-one WVR scenario was selected which would initiate from a neutral starting condition. Since the study was WVR, both pilots would know where the target was at the start of the engagement and would be restricted from launching a missile or firing the gun until after a timer had elapsed. This mimicked a visual identification of an unknown bogey (for both pilots) and neutralized any initial advantages. Unfortunately, these combat artificialities or Rules of Engagement (ROE's) were not enough to neutralize all simulation biases as we will later illustrate.

There were also several other requirements placed on the experiment which impacted its development. The first requirement was that

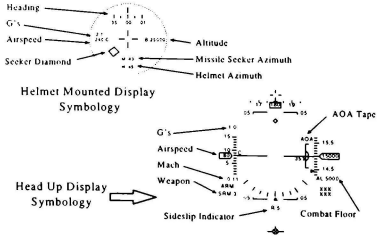


Figure 3 HUD and HMD Symbology

high fidelity aircraft and missile models were to be utilized. The primary reason for this was to validate the MATV/F-16 handling qualities simulation model using MATV program test pilots and flight test data. This requirement impacted the entire simulation from software development to issues involving Pilot Vehicle Interfaces (PVI). The second requirement was that the simulation infrastructure was to be used to investigate avionics system latency and accuracy issues. This implied that the software structure of the HMD, radar, PVI, head down displays (HDD) and missile seeker software had to work in a manner similar to an actual aircraft. The last major requirement was that the PVI and displays used for the MATV and "Look and Shoot" programs would be used with little or no modifications except in those cases where a conflict occurred. The simulation Head Up Display (HUD) and HMD symbology is shown in Figure 2 while the stick and throttle switchology

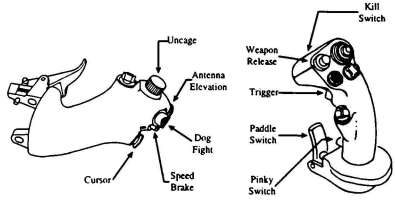


Figure 2 Stick and Throttle Switchology

used is shown in Figure 3.

SIMULATION HARDWARE

The Wright Laboratory Flight Dynamics Directorate simulation facility is structured around a centralized computer deck with two control rooms, manned combat stations, and a high bay containing the Large Amplitude Multi-Mode Aerospace Research Simulator (LAMARS) and Mission Simulator One (MS-1). The LAMARS is a five-degree-of-freedom, motion based simulator consisting of a twenty-foot diameter spherical projection screen mounted on the end of a 30 foot beam. The MS-1 is a fixed-base forty-foot diameter projection dome simulator similar to those in many simulation facilities. The Wright Laboratory facility is designed to conduct either single-ship flying qualities experiments or multi-player

engagement simulations, depending on test requirements.

The LAMARS was configured to replicate the MATV/F-16 cockpit. The cockpit, Figure 4, was configured with an F-16 sidestick and throttle, multi-function displays, instrument cluster, HUD, and Kaiser 12° Field of View (FOV) HMD. Three Liquid Crystal Display (LCD) projectors provided a 120°x40° FOV out-the-window scene. Two calligraphic laser projectors were used to draw images of the target aircraft and missiles. Missile images were only projected during motor burn due to the limited number of laser projectors. A Polhemus magnetic tracker system was integrated with the Kaiser HMD to provide aiming information to the missile seeker.

The MS-1 was configured to represent the threat aircraft cockpit. This cockpit was configured with a 29 inch monitor for displaying cockpit instruments and heads down displays. The HUD imagery was overlaid on the out-the-window visual scenery using a monochrome projector. The front, 180° FOV, out-the-window scenery was projected using one of three light valve projectors. Like the LAMARS, threat images were drawn using calligraphic laser projectors. In addition, an identical Kaiser 12° FOV HMD integrated with a Polhemus magnetic tracker system was used in the MS-1. Polhemus tracker specifications for both the LAMARS and MS-1 are listed in Table 1.

Motion Box	MS-1	LAMARS
X-axis Size	15.0 inches	16.0 inches
Y-axis Size	19.2 inches	16.0 inches
Z-axis Size	9.0 inches	11.0 inches
Positional Accuracy	0.032 inches	
Rotational Accuracy	0.5 degrees	
Update Rate	32.0 ms	

Table 1 - Polhemus 3-Space Magnetic Tracker Specifications

Even with these types of simulators and equipment, several issues arose during testing that would definitely have changed the experiment and the simulation hardware if they had been known in advance. Since this was a one-versus-one experiment, the pilots were initialized at opposite headings at a position suitable for entry into a single circle or two circle fight. During operation, the pilots visually tracked one another in an attempt to gauge what the other would do upon entering the engagement. The problem in the LAMARS was that the visual system FOV was too small. Though extending out to 60° from center, many pilots found that this FOV was not adequate and caused them to roll ratchet the airplane or lose situational awareness in the turn while they attempted to keep their eyes on the target.

Another problem encountered was with the choice of target projectors. Although the calligraphic projectors were very bright and provided enough detail to discern between different model types, these images did not provide the correct type of cueing for air-to-air combat. Pilots found it difficult to judge target

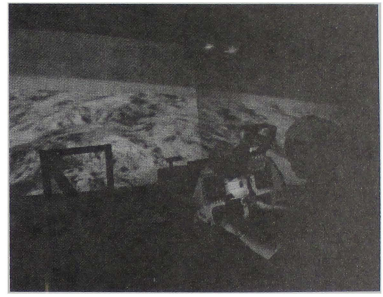


Figure 4 LAMARS Cockpit

orientation and aspect angle from the projected laser image. In fact, pilot comments indicated that the only real useful information obtained from the laser images was azimuth and elevation. A few pilots commented that they would rather have had an image that looked less like a specific aircraft but provided them better maneuver and aspect angle information.

The HMD and magnetic tracker system also impacted pilot performance. The mapped magnetic tracker motion box for both systems was set up around the design eye for each simulator cockpit. Despite this, the size of the motion box in both simulators was still too small. During the engagement, pilots would lean their heads back as far as possible to get the highest elevation angle for lock-on during a turn. Unfortunately, this tended to put the tracker receiver outside the allowable motion box

volume causing tracking errors during target acquisition. Consequently, pilots had to monitor their head position during tracking to avoid moving their heads outside the tracker volume during acquisition. Unfortunately, this problem did not arise during development and shakedown testing. During development, we were more concerned with azimuth accuracy at around 90° off-boresight. It was also difficult to get one's head back far enough to achieve 60° of elevation for target tracking. However, pilots say they can achieve looking up 90° through the canopy while under g-load! How this may have affected the results is unknown, but a larger motion box volume is something that will have to be investigated.

SIMULATION SOFTWARE

Initially, the simulation was required to use the highest level of model fidelity possible for the major simulation components: missiles, radar, and aircraft. However, during the course of the simulation's development, some trade-offs had to be made and a few modeling inaccuracies were encountered.

The requirement for a high-fidelity missile model resulted in an attempt to integrate the non-real-time Trajectory Analysis Program (TRAP)[8]. TRAP is a missile model development and analysis environment used by both government and industry to estimate missile performance in 1-on-1 engagements. However, several problems were encountered which are excellent examples of the types of problems one can expect when attempting to integrate "legacy" packages such as TRAP into a real-time environment.

One of the major problems encountered was the execution rate of the TRAP based missiles. For our TRAP models, a frame rate of 10 ms was required. Due to the inability to run the TRAP routines at this frame rate real-time, studies were done to see the affect of executing the missiles at the simulation frame rate of 20 ms. When comparing the results, the differences in the majority of missile parameters such as time of flight and gimbal angles were very small. However, the parameter that was most affected was miss distance. For the slower update rate, miss distances were sometimes significantly greater than the faster rate even though the outcome (hit or miss) was always the same. Since probability of kill (PK) tables were not utilized in this study, a target would always be dead if it was within the lethal radius of the missile. However, this outcome would not have been the same if PK tables were being utilized and, in fact, a target would still be alive if the miss distance were close to the lethal radius of the missile.

Another equally important problem was the desire to run multiple missile flyouts during an engagement. Unfortunately, the TRAP software was not designed to model more than one missile per engagement. A special missile interface routine was written so that multiple copies of TRAP could be executed on separate processors. This increased the number of missiles that could be in the air at one time to four. Modifications were also made to the parameter initialization routines so that they were called only once during task activation to make them more suitable for real-time operations. Even with these software modifications and executing each missile on a dedicated CPU, it was still difficult to meet frame time requirements. As a result of these problems, all work with TRAP was abandoned and an alternative missile software package was obtained.

The alternative missile software package was a medium-fidelity, five-degree-of-freedom (5-DOF), point-mass model that emulated the required critical missile elements. The final model used aerodynamic tables, motor burn profiles, time-of-flight limits, and linear guidance laws. In addition, the missiles were integrated with a seeker that could be slaved using either the radar or helmet. The seeker model used information such as gimbal limits, slave limits and slave rates for seeker dynamics and target aspect angle, airspeed, altitude and throttle for infrared signature and signal-to-noise ratio calculations. The final model was of course a trade-off between fidelity and performance. Launch envelope charts compared well between the actual missile and the 5-DOF model. However, for our studies, the critical components of the missile model were seeker characteristics, PVI interfaces and flyout agility. Warhead performance factors and endgame performance were not as critical since the tests were considered to be well within missile limits.

Since the experiments were WVR, engagement outcome was also dependent on aircraft performance, flying qualities, and agility. In

addition, the desire to use pilots who had flown the MATV aircraft drove the selection of a six-degree-of-freedom handling qualities (HQ) model with a detailed aerodynamic database and flight control system. However, unexpected problems did arise with this high-fidelity aircraft model. It was found that the engine model was inaccurate at high angles of attack. Using inputs such as throttle setting, Mach, and altitude, the software calculated thrust via interpolation through engine thrust tables. This type of engine model is more than adequate for many flying qualities experiments and has been used in the past without any problems. However, since no thrust losses occurred with angle of attack, an unrealistic pitching moment was generated during high-alpha excursions that permitted unrealistic maneuvers. It wasn't until a chart was obtained detailing engine thrust loss with angle of attack that the problem was rectified (Figure 5).

Another drawback found with the high-fidelity aircraft model is still being worked. Major differences were uncovered when comparing charts of aircraft performance based on the aerodynamics of the HQ model versus drag polars from a performance database. The

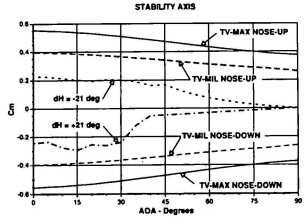


Figure 5 Cm Due to Thrust Vectoring and Control Surfaces

aircraft model documentation did state that drag was only accurate at certain points and therefore should not be used for performance calculations. Because of this, the turn performance and energy maneuverability of the HQ model deviated from standard performance plots more than expected raising the issue of just what does one mean by "high-fidelity". This implied that the high-fidelity handling qualities model was actually of lower fidelity in terms of turn performance and energy maneuverability.

The radar model also had its share of compromises. The test matrix included blocks which did not include the use of an HMD by one of the aircraft. Because of this, a functional radar model was required and software from a previous program was implemented. The radar software used radar cross section tables and radar performance parameters to determine signal-to-noise (S/N) ratios using the classic radar equations. Doppler notch effects were implemented as velocity checks based on look-down angle and the difference between ownship and target inertial velocities. Look-down clutter was modeled as a simple percent degradation in S/N ratio. Table 2 lists the input parameters required to specify the radar's performance.

Radar Input Parameters	Units
Receiver Noise	10^(db/10)
Receiver Temperature	Kelvin
Gain	db
Bandwidth	Mhz
Peak Power	Watts
Low PRF	Hz
Medium PRF	Hz
High PRF	Hz
Pulse Width Low PRF	Seconds
Pulse Width Medium PRF	Seconds
Pulse Width High PRF	Seconds
Main Lobe Loss	%
Wavelength	Meters
Beam Width	Degrees
San Rate	Degrees/Sec
Swerling Uncertainty Table	None

Table 2 Radar Parameters

The radar model also had three basic scan patterns or modes. The first pattern was the standard rectangular, 4 Bar x 60°, track while scan mode that was slewable in elevation. The second pattern was a 10° x 60°, body fixed air combat mode. The third was a rectangular, 4 Bar x 60°, pattern that was slewable in both azimuth and elevation using the throttle cursor. Many of these patterns had to be added and PVI integrated based on pilot comments during shakedown testing. It was found that, depending on the pilot's background, certain radar modes were expected by certain pilots. It took the negative comments from several pilots to finally convince the engineers working on the software that the modes had to be added. Rarely in testing did the pilots use any mode other than the 10° x 60° body fixed pattern but their acceptance of the simulation was dependent on the availability of the other modes. Fortunately,

all of these elements, along with a radar's head down display provided the evaluation pilots and the test a radar model suitable for the experiments.

Finally, it should be noted that a fire control system for generating launch envelope information was attempted but not utilized. Problems were encountered finding suitable software since developing one from scratch was too time consuming. Once again, an attempt was made to use legacy model software. A method was found but proved to be too complex and time consuming and was therefore delayed for future application. Fortunately, the impact on the study of no fire control software was minimal due to the WVR conditions. However, the complexity of fire control software should not be underestimated. Batch simulations can use fast fly-out techniques because they are not running real-time. Real-time simulations, on the other hand, must run the task in a slower cycle to use fast fly-out techniques or develop other schemes such as lookup tables or special functions. Therefore, fire control requires far more effort than one might expect.

PILOT TRAINING AND PVI

Pilot training is required before any simulation testing can occur. For handling qualities experiments, this training tends to be short (half hour to an hour) and usually precedes regular testing. This is because handling qualities experiments typically use test pilot school graduates familiar with flying qualities evaluation methods and are familiar with the Cooper-Harper and Pilot Induced Oscillation (PIO) Rating Scales. In addition, many test pilots tend to be more "forgiving" towards engineering simulations in which the cockpit layout and displays may not be as realistic or representative of the aircraft they usually fly. Flying qualities experiments also tend to be stick and rudder evaluations and not reliant on other major aircraft subsystems for the experiment.

Pilot training time for mission effectiveness combat studies like the Wright Laboratory Flight Dynamics Directorate simulation take far longer than one might expect. The time required to train pilots to fly these types of simulations tends to be two hours or more with additional time for ground school and background briefings. A perfect example of this is the simulation training time required for the Multi-System Integrated Control (MuSIC) Program which was 20 hours prior to a test session[9]. A major reason for the long training length stems from the technologies being used and tested. New displays, switch functions, and tactics all require time to learn so that a pilot can use them effectively in combat. Even displays not used during an experiment can take valuable simulator time. In our experiments, pilots had to learn how to boresight or initialize the HMD/Tracker system that required the use of an unfamiliar display in the HMD. If the boresight procedure was performed incorrectly, HMD and missile seeker performance would be degraded and prove to be a detriment to the test and increase the frustration level of the pilot.

Even with time allocated for training, pilot frustrations still occur. A major cause is PVI differences between what a pilot is used to flying and what is currently mechanized. Anyone attempting to mechanize the PVI of an aircraft based on the information in a Technical Order for that aircraft will understand this problem. In addition, aircraft are constantly being updated which introduces another source of potential differences between the aircraft and the simulation. Questions such as "Why isn't this radar mode mechanized? I use it all the time!" are sure to arise and sometimes degrade a pilot's acceptance of a simulation and the quality of the data he or she might provide.

However, this does not mean that pilot comments on training or PVI are always negative. In fact, pilot insight is generally better than that of the engineers putting together the simulation. The selection of one switch over another may seem intuitive to a engineer and make the task easier while missing another potential conflict. The addition of PVI experts and early pilot inputs into a simulation should be a definite goal of any combat simulation effort.

TEST SETUP

Design of Experiment issues and the development of a test plan are two interrelated issues which can have as much an impact on simulation results as software fidelity. Obviously, the key to any successful experiment is the quality of the data it generates. Achieving this during manned simulation tests is extremely difficult due to the large number of variables that are beyond the control of the simulation engineer. This includes things such as simulator

biases, pilot experience, and run length. In addition, the financial cost of conducting manned simulations requires sacrifices in both test complexity, run length, and sample size in order to keep it at a minimum.

Design of Experiment (DOE) methodologies are not as widespread as they should be and many engineers are just beginning to use them. DOE techniques are also designed for more industrial applications, requiring a bit more imagination in their usage. Determining the DOE interactions can also be frustrating and very difficult since many aircraft technologies definitely "interact" with one another. A prime example from this experiment would be the HMD and high off-boresight missile seeker. DOE techniques are also designed for studies in which a test or experiment can be run repeatedly with little or no variations between runs except for the experiment variables. Unfortunately, pilots have different techniques and experience levels requiring the test engineers to slightly modify the DOE process. In addition, pilot learning curve effects and fatigue must also be considered.

Pilot variability is a major factor in conducting and analyzing simulations. Flying qualities researchers have defined two forms of pilot variability as intrapilot and interpilot respectively. Intrapilot variability is defined as the pilot performance variability between tests of the same configuration. Interpilot variability is defined as the performance variability between pilots of a single aircraft configuration[10]. The number of pilots to decrease interpilot variability as well as provide adequate numbers for the desired confidence levels, is impacted by intrapilot variability. Though flying qualities experiments have the added difficulty of using the Cooper-Harper scale, intrapilot and interpilot variability are still major factors during combat simulations. Pilot performance, adaptability, biases, and background may affect combat simulation results even more due to the desire to use line pilots instead of test pilots. Therefore, more caution is required when creating combat simulation test matrices, analyzing the data, and determining the number of required pilot subjects.

The Wright Laboratory combat study attempted to take as many of these factors into account. The abbreviated test matrix is shown below in Table 3. There were initially eight initial conditions (I.C.'s) which represented different slant ranges and starting azimuths and elevations of the adversary aircraft with respect to the blue aircraft. During shakedown testing, five of these conditions were dropped. Initially, it was believed that the different initial condition starting points would cause different maneuvers to be flown and also provide a "long-look"[11] technique for the pilots to make an evaluation. However, executing this larger matrix during shakedown proved to be too time consuming and with minimal benefits. During the first round of testing, it was also decided to drop the 10K altitude portion of the matrix because it added a large amount of additional time to the test matrix. Pilots also commented that the test condition was unrealistic from a combat standpoint. However, the trade off one must make is whether or not the test point is 'realistic' from an engineering test standpoint.

During testing, three groups of two pilots each were flown with each duo of pilots flying in both cockpits. Results of each test are still being analyzed using different means of averaging pilot results in an attempt to reduce interpilot variability and simulator biases. Ideally, all pilots in a study should fly against one another to obtain all pilot combinations for removal of biases. The test matrix should also be randomized in order to remove learning curve affects which can give the data phantom trends.[7]

Starting Conditions	Thrust Vectoring				No Thrust Vectoring			
Altitude: 25K	HMD		No HMD		HMD		No HMD	
Mach : 0.8	Misl	Misl	Misl	Misl	Misl	Misl	Misl	Misl
I.C.'s : 1-3	A	B	A	B	A	B	A	B
Starting Conditions	Thrust Vectoring				No Thrust Vectoring			
Altitude: 10K	HMD		No HMD		HMD		No HMD	
Mach : 0.4	Misl	Misl	Misl	Misl	Misl	Misl	Misl	Misl
I.C.'s : 1-3	A	B	A	B	A	B	A	B

Table 3 Test Matrix

Another interesting issue that was encountered was the determination of the "baseline" configuration. Does one want to set the "baseline" configuration to current technology and what pilots are used to using? Or, should the "baseline" be the configuration with all the "best" technologies with current technology treated as a "degradation". Pilots generally are

trained with the most advanced technology combinations and are already familiar with current technology limitations. It was found that pilots would have preferred to have the advanced technologies first with other combinations treated as degradations. This approach may also help minimize learning curve affects in the test data.

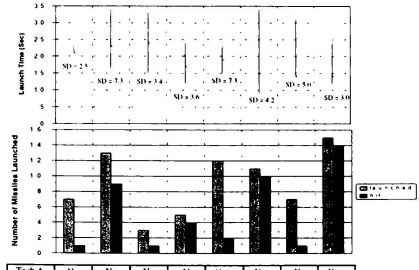
DATA ANALYSIS

The recording of simulation data takes several forms and includes continuous, event, verbal, and written (pre- and post-simulation pilot questionnaires). Data recording is one area where it does not pay to be efficient because Murphy's Laws are in full affect. Any attempt to be efficient results in eventually needing data from parameters you did not record. In addition, many continuous parameters may be repeated as events for rapid correlation with other actions, such as missile launches or mode changes. Written questionnaires are also important sources of data because they can help organize comments made during the test and focus pilot remarks afterwards.[12]

One major problem that was encountered was determining adequate measures of performance and effectiveness. Measures of Performance (MOP) are indicative of system's activity during an experiment, such as the number of switch actions or time to launch. Measures of Effectiveness (MOE) are indicative of a system's performance in achieving an operational goal, such as aircraft loss rates, exchange ratios, or number of threats destroyed.[13] Terms such as exchange ratio, however, are generally suitable for many-on-many air combat simulations or simulations in which probability of kill (P_K) and survival (P_S) tables are used. For the Wright Laboratory tests, a missile kill was achieved as long as the target was within the lethal radius of the warhead and arm time had been exceeded. This inhibited using a term such as exchange ratio as an MOE because many runs ended as a mutual kills and no increase or decrease in effectiveness could be demonstrated. Therefore, a different method of demonstrating mission effectiveness had to be created.

Figure 6 is a First Launch Summary Chart developed for demonstrating the effectiveness of the different technology combinations. The chart is a combination of two separate charts displayed together to provide the whole picture of how a particular technology combination performed. The top half of the chart consists of missile launch times showing low, mean, and high times of all first missile launches for the blue aircraft. These times were extracted from all pilots flying that particular technology combination. Attached to each low-mean-high bar is the standard deviation of all launches which provides an estimate of distribution. The lower bar chart shows the number of missiles launched and the number of missiles achieving a kill. This illustrates a missile's failure due to seeker limitations, tip-off (missile aligning with relative wind), and lack of performance. The bar chart x-axis lists the corresponding technology combinations.

Though the chart is rather cumbersome, it does illustrate several important factors in increasing mission effectiveness in the one-versus-one scenario for a particular technology combination. These factors are launch time, number of missiles launched, and the number of successful missile hits. The number of missiles launched and launch times captures system avionics, missile, and aircraft performance influences on the engagement while the number of successful missile hits captures missile flyout and endgame performance. This complex chart is considered a better choice than many past attempts to capture weapon system effectiveness via a new MOE.



Tech A	No	No	No	No	Yes	Yes	Yes	Yes
Tech B	No	No	Yes	Yes	No	No	Yes	Yes
Tech C	No	Yes	No	Yes	No	Yes	No	Yes

Figure 6 First Launch Summary Chart

The problem in determining suitable MOEs and MOPs will always be difficult. A major problem is conveying to the audience the statistical relevance and confidence level that exists in the

data. Conveying this information in a clear and simple manner is a challenge. Also, comparing apples, oranges, and carrots, (as was done when looking at the synergy issues of thrust vectoring, HMD's, and high off-boresight agile missiles) can be difficult during data analysis. MOEs such as exchange ratio tend to mask details and are unsuitable for conducting HMD evaluations or PVI issues which can impact performance in detrimental ways that are hard to quantify. Therefore, test matrix design and data analysis require careful attention and blindly using traditional MOE's should not be done.

CONCLUSIONS

Developing, conducting, and analyzing data from multi-player combat simulations is extremely difficult and requires extra vigilance on the part of the simulation and test team. Simulation facilities and groups are being asked to analyze ever more technologies due to increasing development and test costs. In order to make informed decisions and investments, bad simulations must be avoided so erroneous results are not given which will undermine the confidence in future simulation results. Adequate model fidelity, documentation, and test methods, are required if future simulation studies are to maintain a high level of confidence in their results. Already, many groups and individuals are beginning to doubt the veracity of simulation results due to poor documentation, fidelity, and test methods. There is definitely a difference between engineering simulations and 'simulations'. Unless great care is taken during the development of a simulation, time and effort will be wasted in developing a "video game" which does not achieve a goal.

REFERENCES

1. Bent, N. E., et al. *The TAC BRAWLER Air Combat Simulation Analyst Manual (Rev 6.1)*, Decision-Science Applications Report #906, May 1993
2. Kish, Brian A, et al. *Practical Limits of Supermaneuverability and Full Envelope Agility*, AIAA-96-3493-CP, July 1996
3. Woodfield, A.A., et. al. *Flight Mechanics Panel Working Group 16 on Aircraft and Sub-system Certification by Piloted Simulation,* AGARD-AR-278, September 1994.
4. Lynch, Urban H.D., et al. *Tactical Evaluation of the Air-to-Air Combat Effectiveness of Supermaneuverability*, WRDC-TR-90-3035, June 1990
5. Kidman, David S., et al. *Evaluation of the F-16 Multi-Axis Thrust Vectoring Aircraft*, AFFTC-TR-95-12, September 1995
6. Hughes, David "Off-Boresight Missile Test Highlights U.S. Capability." *Aviation Week and Space Technology*, 6 June 1994
7. Riley, David R. *Simulation Requirements for the Assessment of Enhanced Agility*, SAE# 91-1131, Presented at the SAE Aerospace Atlantic Conference April 1991
8. Byram, Timothy R., et. al. *Software User's Manual for TRAP 3.1*, Contract No F33657-88-D-0076 Final Report, 31 March 1993
9. Doane, P.M., et. al. *Multi-System Integrated Control (MUSIC) Program*, WRDC-TR-90-6001, June 1990
10. Wilson, David J. and Riley, David R. *Cooper-Harper Pilot Rating Variability*, AIAA-89-3358-CP
11. Wilson, David J. and Riley, David R. *More on Cooper-Harper Pilot Rating Variability*, AIAA-90-8922
12. Master, Russell M., et. al. *White Force Employment in T&E Programs - A Test Director's Handbook*, Informal Technical Report, December 1987
13. Lehman, Edward F. and Jenkins, Mary-Louise *Handbook For Conducting Pilot-in-the-Loop Simulation for Crew Station Evaluation*, HSD-TR-90-007, November 1989

MODELING THE DYNAMICS OF UNDERSEA VEHICLES DEPLOYED FROM SURFACE/AIR PLATFORMS

S. A. Moyer†
Naval Undersea Warfare Center Division, Newport
Newport, RI 02841
and
K. W. Watkinson††
Vehicle Control Technologies, Inc
Reston, VA

ABSTRACT

Undersea vehicles and weapons that are deployed from above the surface, via tube launch or air drop, experience severe dynamics during the launch and water entry process. These dynamics may impact the ability of the vehicle to carry out its intended mission. In the past extensive at-sea and air launch testing took place to ensure vehicle survivability and performance. The current effort examines launch and water entry models developed in order to respond to reductions in testing. The basis for the models is discussed and simulation output is compared to launch and at-sea data.

NOMENCLATURE

A_i	= Influence coefficient
A_{eff}	= Effective area, ft^2
BL	= Aircraft Buttline, in
$C^*_{D_c}\|_{max}$	= Peak drag coefficient at water impact
C_N	= Normal force coefficient
D_{prop}	= Propeller diameter, ft
FS	= Aircraft Fuselage Station, in
J	= Propeller Advance Ratio
n	= Frequency of propeller rotation, rpm/60
r	= Weapon radius at point of constant cross section, ft
U	= Velocity, ft/sec
WL	= Aircraft Waterline, in
α_{xy}	= Local flowfield sidewash angle, deg
α_{xz}	= Local flowfield upwash angle, deg
ν	= Second Modified Advance Ratio of the propeller
ρ	= Density of medium, slug/ft^3
θ	= Entry Angle, deg (vertical entry = 90 deg)

---

† Mechanical Engineer, Member AIAA

†† Vice President

This paper is declared a work of the U.S. government and not subject to copyright protection in the United States

INTRODUCTION

Successful deployment of undersea vehicles from surface and air launch platforms presents many unique challenges. Such deployment occurs from vastly different platforms (fixed wing, helicopter, surface ship), imposing vastly different conditions at water entry. The water entry process imposes extremely high forces and resulting dynamics on the body, as compared to the remainder of the mission. Often, at this same time, the vehicle is establishing propulsive power and just beginning to exercise control.

In an era of reduced at-sea testing it is desirable to investigate many aspects of the deployment of these vehicles via modeling and simulation. Modeling the resulting body dynamics associated with such deployments can be divided into two primary regimes: the time from launch until water impact and the time from water impact until a controlled pullout has been achieved. Prior to water impact, the purpose of the model is to determine the initial conditions at water impact which will determine the resulting water entry dynamics. After water impact the model must include the effects of impact shock, cavity formation, cavity closure, engine transients and the onset of control. This paper describes models developed to support these launch, water entry and pullout phases. A comparison to actual launch and water entry data for a test vehicle is presented.

MODEL DEVELOPMENT

Background

The Naval Undersea Warfare Center Division Newport (NUWCDIVNPT) maintains a suite of undersea weapon six-degree-of-freedom (6DOF) Vehicle Dynamics Simulations (VDS). The simulations have historically focused on the undersea behavior of these vehicles. The NUWCDIVNPT VDS combine a standardized equations of motion representation along

with vehicle specific hydrodynamic coefficients, autopilot, propulsor, actuator and sensor models. The equations of motion formulation for these vehicle simulations are described in reference 1. The modular approach of the general suite of simulations is shown in Figure 1. Recently the performance of the vehicle during water entry has received increased interest and a requirement for higher fidelity model has resulted in the effort described here.

Launch

A launch model for an undersea vehicle should provide an accurate representation of the vehicle state at the time of water impact/entry. This forms the set of initial conditions for the water entry model. A launch model is also instrumental in determining air stabilizer designs, launch envelopes and tube launcher settings that will support acceptable pullout and component survivability. Both air launch and tube launch models have been incorporated as part of this effort.

The air launch model in this effort is divided into three main phases. The first phase is for internal bay launches only and is considered to be the time prior to the vehicle exiting the bay. The second phase is that period of time when the vehicle is in the influence of the aircraft flowfield. The final phase is after air stabilizer deployment. A six-degree-of-freedom trajectory calculation code based on body loads is applied to determine the trajectory time history of the weapon from release until water impact. It is the determination of the loads and accelerations where the three phases differ. Due to the relatively low air speeds of the launch platforms from which these sorts of vehicles are typically released, the vehicle experiences very low aerodynamics forces while it is in the bay. Therefore, the current model uses a purely inertial model while the vehicle is in the bay. If high-speed bay releases were to be considered this simplification would limit the usefulness of the approach.

When the vehicle is in the influence of the aircraft flowfield, the air launch model is based upon a superposition method developed by the weapon separation community known as the Influence Function Method (IFM) (Reference 2-4). The IFM provides a method for determining vehicle loads in a non-uniform flowfield such as the flow field around the launch aircraft. It uses a description of the local flow angularity near an aircraft and a predetermined correlation of local flow angle to a body segment force and moment to calculate total forces and moments of a body in a flowfield. For a particular coefficient, for instance the normal force coefficient, C_N, the IFM is applied as indicated in Equation 1, below:

$$C_N = \sum_{i=1}^{N} \left[A_i \alpha_{xz_i} \right] + C_{N0} \qquad (1)$$

where A_i is the predetermined correlation or 'influence' coefficient , α_{xzi} is the local upwash flow angle, and C_{N0} is any force associated with zero flow angle. The summation is carried out over an N-segmented body. There are coefficients equivalently defined for pitching moment (also dependent on α_{xzi}), and side force and yawing moment (both dependent on α_{xzi}). The determination of the influence coefficients is a key to successful application of this process and well suited for the application of computational aerodynamics tools (Reference 5). This approach was used during the current study to determine the influence coefficients for several air launched torpedoes. The IFM can also be applied in an incremental sense if higher fidelity (uniform flowfield) aerodynamic data is available. In this case the IFM is used to calculate a delta between the IFM calculated non-uniform loads (α_{xzi} and α_{xzi} are vary) and the uniform loads (α_{xzi} and α_{xzi} are constant). This delta is then applied to the higher fidelity aerodynamics.

The other essential element to the application of the IFM is the determination of the local flowfield. Both wind tunnel and computational methods have been used in the determination of flow angularity around aircraft. Potential flow solutions have been shown to give accurate descriptions of aircraft flowfields (Reference 6-7). In the current effort a model of the S-3 was constructed and analyzed using the PAN AIR linear potential code (Reference 8). The model of S-3 is shown in Figure 2. This model was then used to calculate the flow angularity under the weapons bay of the S-3. Figure 3 depicts the area of interest under the S-3 weapon bay and the points of flow survey. Figure 4 presents the representative flow angularities for a particular station under the bay at one flight condition. Of particular interest is the variation in side flow close to the fuselage, it is clear that depending on where the body is positioned within this flowfield, it will experience vastly different loading. The flow angularities are used in conjunction with Equation 1 to determine the loads as the body traverses through the flow field. Such a model not only provides the necessary modeling of the vehicle for purposes of progressing to the next phase (stabilizer deployment)

but also permits analysis of any safe separation concerns.

Once the air stabilizer deploys, it dominates the vehicle dynamics, and a ballistic model based on drag time histories for the particular air stabilizer is used to determine the trajectory to water impact (Reference 9). The present method combines the three phases and appropriate load calculation scheme into a single air-launch trajectory model. A representative comparison of the outputs of the simulation to flight test data is presented in Figure 5 for down range travel, time of fall and entry angles. Each case in Figure 5 represents a different set of launch conditions (airspeed and altitude). The final result of the model is a set of initial conditions of the weapon at the time of water impact which then can be used to initiate the water entry model. This model also will identify what flight envelope is achievable based on limits on the water entry initial conditions that may exist for vehicle or component survivability. In the context of an integrated simulation or engagement model this model could be used to develop a matrix of entry conditions keyed to launch platform conditions.

For surface ship launches, a pneumatic tube launch model previously developed by the Naval Underwater Systems Center (Reference 10) is used. The model represents the gas dynamics process when the stored energy of a pressurized sphere is released and transferred to the vehicle. The model incorporates frictional and blowby losses in order to give an accurate tube exit velocity. Reference 10 has previously demonstrated the accuracy of this model.

Water Entry

There exists a complex sequence of events that occur when a body impacts and enters the water from the atmosphere. First, the body experiences an impact or shock phase, which is of extremely short duration, but can impart a significant force to the vehicle. The body then progresses through a deceleration phase as the flow patterns are established, to a phase where the body is in a cavity vented to the atmosphere. Eventually the cavity will close at the surface, then collapse down the length of the body until the body is fully wetted. The actual response of the body is highly dependent upon the shape of the body and its initial conditions at impact. The physics of water entry have received much attention over the years and detailed descriptions, both quantitative and qualitative, of the preceeding events have been offered (Reference 11). The current effort centers around determining the critical physical quantities that influence the ultimate dynamics of the vehicle for each of these phases.

For the purposes of modeling the vehicle dynamics it is not so much necessary to rigorously describe the full water entry physics, but rather to quantify the resulting effects on the vehicle. For flat nose bodies typical of undersea vehicles, the force of primary interest is drag. For other shapes, the non-axial forces acting on the nose at entry cause a phenomena known as whip which invalidates this simplification. Even flat faced bodies, if the entry angle is sufficiently shallow, could also experience whip, and the models developed here do not apply to these conditions. It has been assumed in the formulation of these models that the vehicles are entering the water at angles sufficiently large to avoid significant whip forces. Whip is one contributor that can lead to weapons broach. Other contributors to broach are insufficient control authority, inadequate control system response, or motion measurement errors during the large transient motion conditions occurring during water entry. These other broach contributors are modeled in the simulation. Whip forces are investigated using sensitivity analyses.

Each phase of the water entry process was analyzed and the extent to which modeling is required was determined. To this end, the complex physics of the nose impact shock structure between the air and water interface need not be fully modeled. Modeling the impact phase is reduced to determining the impulsive force that occurs during this brief phase. Empirical data has been used to obtain the impact drag coefficient, predict the peak shock and the overall change in state of the vehicle due to the impact. It was shown during this effort that the peak drag value of flat nosed vehicles at nose impact is related to the entry angle as shown in Equation 2, below:

$$C^{*}_{D_c}\big|_{max} = e^{0.8562 - 6.3894\theta + 16.8296\theta^2 - 15.6549\theta^3 + 5.1942\theta^4} \quad (2)$$

The peak shock can then be determined by :

$$F_{shock} = \frac{\rho}{2} U^2 A_{eff} C^{*}_{D_c}\big|_{max} \quad (3)$$

Calculation of shock is valuable primarily to analyze vehicle and component survivability. The actual change in state of the weapon is effected by the impulse imparted to the vehicle during the shock and flow forming stage. This occurs so rapidly that it is sufficiently represented by an instantaneous change in

velocity. The impulse does not vary significantly with impact angle, but instead is related to the acceleration of the fluid at the nose/water interface up to the vehicle speed. This can be modeled as an added or virtual mass effect over a very short period of time. The added mass associated with this impulse is a function of the geometry of the nose. A sufficient representation of the added mass is:

$$m_{a_{impulse}} = \rho r^3 k \qquad (4)$$

The shape factor, k, would be 1.0 for a sphere and 1.3 for a disk. Most undersea vehicles would be somewhere in between. The actual change in state due to this impulse, though not negligible, is small with respect to the overall deceleration of the vehicle in the cavity running phase.

After impact, the model transitions immediately to a representation for a cavity running body. When cavity running, the force of most importance to the body is the drag. This cavity running assumption is carried until the cavity begins to close over the body. The steady cavity-running drag coefficient for a disk is given in Reference 11 as 0.815. The force on the body during this phase can be obtained from Equation 3 by substituting the cavity running drag coefficient for the impact drag coefficient. The cavity running assumption is a simplification in that it ignores moments generated on the nose due at entry and by the tail impacting the cavity walls. What the model yields is a nominal trajectory which will 'average' the trajectory of many launches. The previous assumption that the vehicle is entering with conditions that preclude whip supports the use of a cavity running assumption or that tail slap is oscillatory, yielding a nominally straight trajectory.

The physics of cavity formation, cavity shape, pullaway and closure are also complex phenomena. However, since the subject of the model is the vehicle and not the cavity per se, the analysis of the cavity can be simplified. Reference 11 offers a description for the shape of an ideal elliptical cavity. It is assumed such a cavity is formed upon water entry and the shape of the cavity is continuously checked to determine at what time the cavity begins to close at the tail. The time from closure at the tail to fully wetted flight is extremely short. During this time the forces and moments on the body are calculated assuming fully wetted hydrodynamics in all axes and decaying the cavity-running drag to zero in proportion to the length of body in fully wetted flow. Once fully wetted flow is obtained, the forces and moments on the vehicle are obtained with the standard VDS hydrodynamic coefficients and equations of motion as developed in Reference 1.

Engine/Propeller Transients

During water entry, after cavity collapse, a vehicle with open bladed propellers is likely to experience dynamics due to flow over the propellers (windmilling), prior to actual engine start. It is conventional to express the forces and moments generated by propellers based on the propulsor advance ratio:

$$J = U/(n * D_{prop}). \qquad (5)$$

It is clear that if the propeller is initially at rest, that this quantity is undefined. The current model uses a previously developed (Reference 12) representation of propulsion dynamics known as the second modified advance coefficient. The second modified advance ratio is calculated as follows:

$$\nu = \frac{nD_{prop}}{\sqrt{U^2 + n^2 D_{prop}^{\ 2}}}. \qquad (6)$$

This new formulation allows an alternate parameterization of propeller thrust (or drag) and torque related to vehicle velocity and shaft RPM that is always defined.

Test data for the water entry of a vehicle equipped with a counter rotating propeller was used to develop windmilling drag and torque versus the second modified advance coefficient. The nature of the resultant drag relationship should cause the weapon to experience deceleration consistent with the test data. The nature of the resultant torque relationship determines the actual response of the RPM time history. A second modified advance ratio model was added to the existing vehicle dynamic simulation for this vehicle. Figure 6 offers a normalized comparison of the model representation of RPM transient upon water entry to a series of in-water data. The in-water data demonstrates the scatter associated with water entry phenomena. However, the model results offer a nominal representation of the dynamics of the propellers during water entry.

MODEL VERIFICATION AND ANALYSIS

Data Comparison

The water entry models described above have been incorporated into an existing NUWCDIVNPT VDS that previously accounted for only underwater, fully-wetted flight. This simulation includes models of the weapon control system/autopilot, propeller, fin actuators and sensors incorporated into the standard equation of motion and modular representation shown in Figure 1. The portion of the water entry trajectory prior to the establishment of fully wetted flow, by its very nature, is extremely short. It is the intent of the models developed during this effort to provide sufficient modeling of the water entry event to provide accurate conditions at the time fully wetted flight begins and existing simulation models are valid.

Data has been shown in the previous paragraphs that verifies that the air launch model and propeller transients are adequately modeled. A series of instrumented at-sea runs of a vehicle undergoing controlled water entry and pullout from a tube launch, was available for verification of the remainder of the water entry trajectory model. No such data was available for water entry for the air launch of this vehicle. In keeping with the objectives for the model, a rigorous verification of the cavity dynamics is not investigated, nor was there proper data available from these tests to do so. Instead the response of the weapon dynamics and trajectory form the basis of the verification.

Of primary interest during the water entry and pullout is the performance in the pitch plane. Figure 7 shows the axial acceleration experienced by the weapon from four in-water test series and the model. The data presented starts (T = 0.0) at the time the vehicle leaves the tube. The in-water data and model both indicate free flight from T=0.0 until just after .6 seconds, at which point water impact occurs. The test data again indicates the scatter associated with the timing and physics of the phenomena under investigation. The comparison indicates that upon water impact, the model predicts the sudden large deceleration due to the water impact and cavity running. Also, the model accurately predicts the effect of cavity collapse on the deceleration process.

Figures 8, 9 and 10 show the same comparison for the pitch rate, pitch angle and depth (depth has been arbitrarily normalized). A consequence of the assumption that only axial force contributions are present during cavity running is evident from these comparisons. While the test data shows an immediate tendency, upon water impact, for the pitch rate to change signs and begin reducing the weapon pitch down angle, the model holds the pitch rate constant until cavity collapse. After cavity collapse the model gives an accurate representation of the pitch rate response. The difference in the pitch response is due to a moment imparted at the nose of the vehicle upon water entry/cavity formation that tends to rotate the vehicle towards the neutral position. This moment is absent in a cavity running approximation and is one reason why the model should be exercised with care at low entry angles, and reinforces the earlier discussion about the use of this model and the development of whip forces at the nose that cause broach. As the entry angles increase, this tendency decreases, and accuracy of the cavity running assumption increases. The current verification was conducted at the low end of the envelope and demonstrates the lower end of entry angle validity for this modeling approach.

Once the proper pitch rate is obtained, at about 1.2 seconds, the remainder of the pullout tracks well with the data. In fact the overall response of the pitch angle (Figure 9) and depth (Figure 10) response are within the scatter of the in-water data. After approximately 2.5 seconds, the control system becomes active and the vehicle is commanded to a pullout depth. Comparison of the model to the test data after this point permits verification of both hydrodynamic and autopilot components of the simulation.

Figure 11 presents the measured roll angle for the in-water data. It is obvious from this data that variation in roll angle is large and no consistent mechanism is responsible for the roll response. The roll response will depend on slight differences due to loading, the launch process, fin alignment, and engine startup to name a few conditions. The best that can be offered, is that rigorous water entry analysis should bound a range of initial roll angles and rates.

Water Entry and Pullout Analysis

The data presented in the previous discussion indicates that for the ship launch case the weapon under consideration had no problems with water entry or pullout. This represents only one possible launch type for these type of weapons. Often such weapons also must be deployed from rotary wing aircraft in hover and both rotary and fixed wing in fly-in conditions. No at-sea testing under these conditions was included in the development of the model examined here. Modeling and simulation was used to make design decisions regarding launch from these platforms.

Often the weapon itself has constraints which limit the water entry initial conditions. These constraints include weapon and component survivability due to shock and the ability to achieve pullout. The combination of weapon release envelope, air stabilizer design and nose protection will determine the extent to which the weapon will survive shock. The vehicle under analysis had certain given shock survivability requirements which translated into an entry velocity vs entry angle envelope. This information was used in conjunction with the air launch model described earlier to establish the release envelope (airspeed and altitude) that resulted in survivable water entry conditions.

The other constraint, successful pullout, must still be examined. Successful pullout is an operational concern. That is, the minimum deployment depth will be determined by the ability of the vehicle to pullout before bottom impact. The most severe initial conditions for pullout are those corresponding to the highest energy entries that the weapon survivability (discussed above) allows. The highest entry angle/highest velocity entry determined from survivability analysis was examined using the simulation. The results of the pitch angle and normalized depth profile are presented in Figure 12.

Whereas the ship launch case indicated a smooth pullout to the commanded pullout depth, these more severe entry conditions result in considerable depth overshoot during the pullout to the same commanded pullout depth. This could possibly present operational deficiencies if the overshoot was beyond deployment depths.

In the ship launch case shown in the verification, it was noted that activation of control was delayed for about 2.5 seconds from launch. This was also the case with the launch shown in Figure 12. This delay is to ensure the vehicle has sufficient velocity and flow over the fins to affect control. The water entry velocities associated with the higher energy, fly in launched should be sufficient to affect control immediately. This being the case the simulation was rerun allowing control as soon as the vehicle is fully wetted. The resulting improvement is shown in Figure 13.

Another alternative is to further reduce the launch envelope to permit entry velocities or angles that permit acceptable pullout. Figure 14 shows the results for a 50% decrease in entry velocity at the same entry angle as Figure 12 and a 75% decrease in entry angle at the same entry velocity as Figure 12. Although this indicates acceptable pullout, it has operational implications in that the release envelope has been further reduced.

Launch from a rotary wing platform in hover poses a different set of concerns. Depending on the altitude of release the air stabilizer may not have time to rotate the vehicle to an acceptable angle to avoid broach. In such a circumstance it is often necessary to rotate the body with some sort of restraint. As previously discussed a vehicle entering the water is likely to have a range of entry angles that are acceptable. The low end will be determined by broach concerns and the upper end will be determined by survivability of the vehicle or components within the vehicle. Assuming a restraint is designed to provide these water entry angles, the model was used to examine the pullout response of the vehicle under analysis. Figure 15(a) shows the pullout response for the lowest allowable entry angle to prevent broach, and indicates good pullout performance. Figure 15(b) shows the pullout response for the upper boundary on entry angle, and indicates an overshoot during the pullout. This analysis indicates that the constraints on the restraint design are more restrictive than the complete entry angle envelope. Further, since the restraint tends to provide a rotation rate and a means of achieving the entry angle, a given restraint will only achieve proper entry angles over a rather small range of hover altitudes.

SUMMARY

Launch, water entry and pullout are critical stages in the deployment of undersea weapons from platforms above the surface. In order to best utilize limited at sea tests, models were developed to allow simulation of the deployment sequence. Existing models, appropriate databases and theory from many sources have been integrated into a modeling approach to accomplish this.

The approach divided the modeling and analysis into two main regimes; launch through water impact and water impact through pullout. Models developed for the first regime concentrated on obtaining an accurate weapon state at the time of impact in order to predict the following water entry and pullout trajectory. The models developed for this regime also yield valuable information into launch envelopes, air stabilizer design and vehicle impact survivability. The launch models were verified to be accurate against test data.

The models for the second regime draw heavily on historical hydroballistics theory and experiment. The purpose of the models in this phase is to accurately

predict the trajectory of the weapon as influenced by the water impact and entry event, coupled with weapon specific models of propulsion and control. Models such as these can then investigate critical phenomena that may impact the overall mission of the weapon. Specific models for an undersea vehicle developed using these approaches were compared to at-sea test data to verify the approach and discuss limitations and restrictions on use of the models. The simulation was then used to expand the limited at-sea testing to launch modes not investigated and make design decisions to enhance mission accomplishment.

REFERENCES

1. Smith, N. S. and Watkinson, K.W., "User's Manual for TRJv: A Six-Degree-of-Freedom Simulation Program for Underwater Vehicles with Significant Vortical Flow Effects," VCT report No. 14, December, 1994

2. Cenko, A., et. al., "Influence Function Predicion of Store Trajectories, Part I Theory and Applications, Part II Description of Computer Code," AFWAL-TR-84-3057, August 1984.

3. Keen, K.S., "A Refined Implementation of the Influence Function and Incremental-Coefficient Methods for Store Trajectory Generation," AEDC-TR-86-35, October 1986.

4. Moyer, S.A., "Investigation of a Flow Angle Based Store Separation Trajectory Program," AIAA-91-2896, August 1991.

5. Morgret, C.H. and Christenson, R.J., "Modification of the Missile Distributed Airloads (MDA) Code to Improved Predictions at Extreme Angles of Attack," AEDC-TR-95-4, May 1995.

6. Moyer, S.A., "PAN AIR Analysis of Simply Connected Control Surface Deflections," AIAA Journal of Aircraft Vol 30 Number 1, January -February 1993, pp 154-156.

7. Cenko, A. and Moyer, S., "Evaluation of the F/A-18E/F Aircraft Flowfield Effect on the HARM Missile," Naval Engineers Journal, November 1994, pp 85-93.

8. Magnus, A.E. and Epton, M.A., "PAN AIR - A Computer Program for Predicting Subsonic or Supersonic Linear Potential Flows About Arbitrary Configurations Using A Higher Order Panel Method," NASA-CR-3251, 1980.

9. McAnelly, L.J., "A CDC Computer Program to Simulate Trajectories of Unguided Missiles," NSWC/DL TN-DK-8/76, March 1976.

10. Elste, C.L. and Clements, M.W., "Computer Simulation of Surface Vessel Torpedo Tube Pneumatic Launch Process", Naval Underwater Systems Center, TM No 31-2141, 1982.

11. Hydroballistics Design Handbook, SEAHAC TR 79-1, January 1979, Vol 1, Chapter 2.

12. Rubis, C.J. and Moken, A.J., "Submarine Propulsion Dynamics Simulation Using a Simplified Steam Generator Model," NSRDC Report 6-207, December 1971.

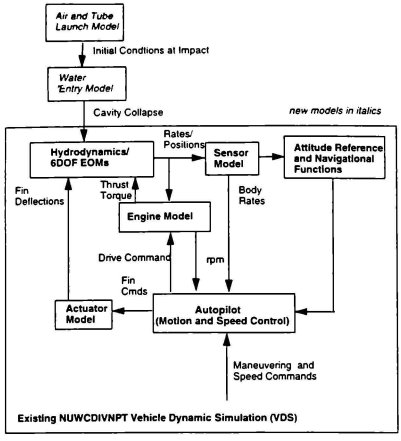
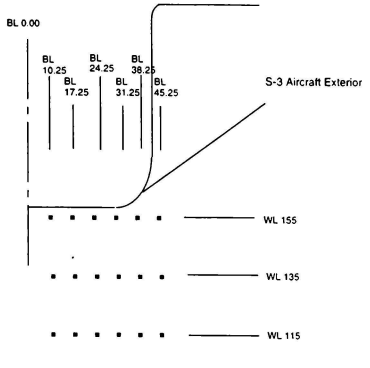


Figure 1 - NUWCDIVNPT Vehicle Dynamic Simulation Block Diagram



Note: For each survey location a longitudinal traverse from FS 250 to FS 500 was made with data at every 10 FS.

Figure 3 - S-3 Flow Survey Coodinates

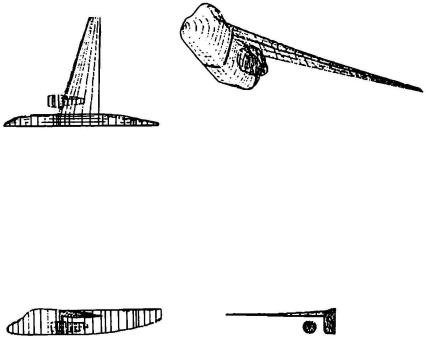


Figure 2 - S-3 PAN AIR Model

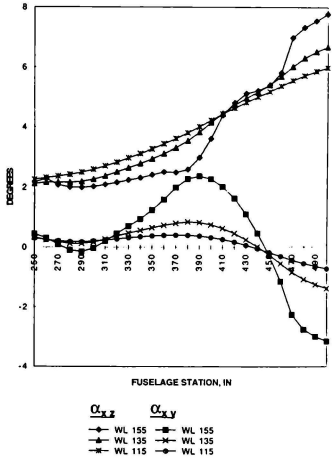


Figure 4 - S-3 Flowfield BL 31.25 M=0.5 AOA=2.0

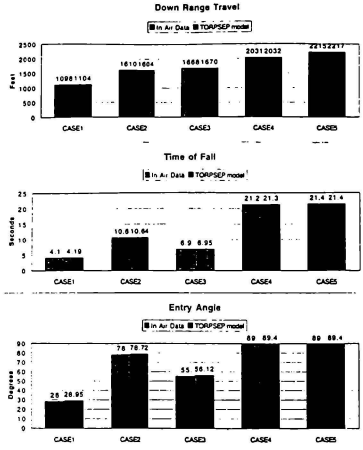
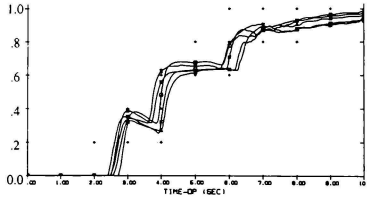
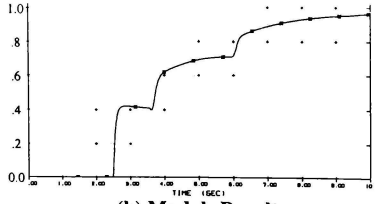


Figure 5 - Comparison of Flight Test Data to Air Launch Model

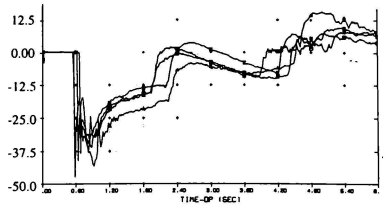


(a) In-Water Test

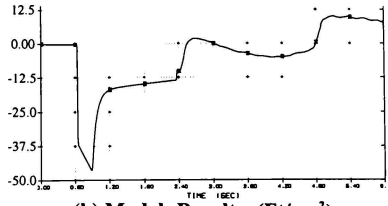


(b) Model Results

Figure 6 - Water Entry Transients of a Counter Rotating Propeller

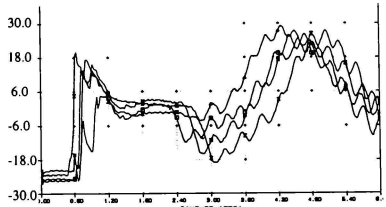


(a) In-Water Test (Ft/sec²)

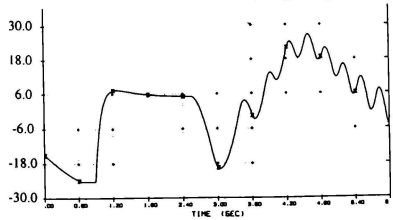


(b) Model Results (Ft/sec²)

Figure 7 - Axial Acceleration, Tube Launch Water Entry

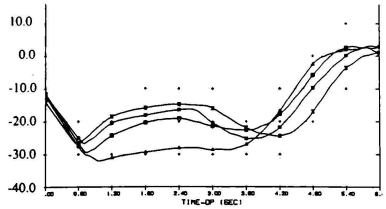


(a) In-Water Test (Deg/sec)

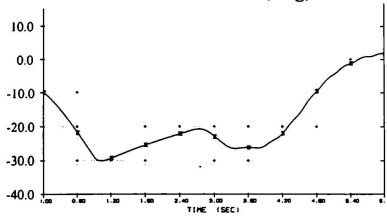


(b) Model Results (Deg/sec)

Figure 8 - Pich Rate, Tube Launch Water Entry

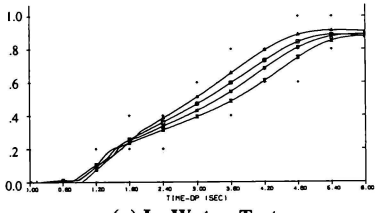


(a) In-Water Test (Deg)

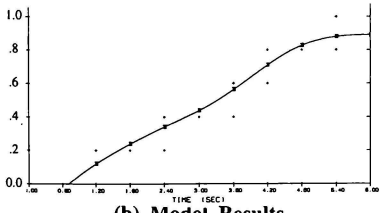


(b) Model Results (Deg)

Figure 9 - Pitch Angle, Tube Launch Water Entry



(a) In-Water Test



(b) Model Results

Figure 10 -Normalized Depth, Tube Launch Water Entry

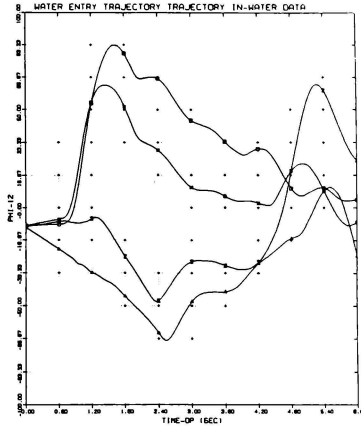
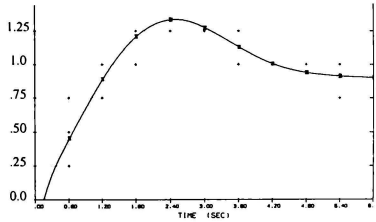
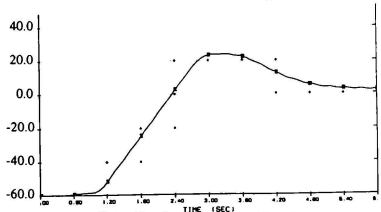


Figure 11 - In-water Test Roll Angle, Tube Launch Water Entry

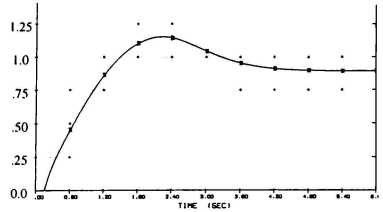


(a) Normalized Depth

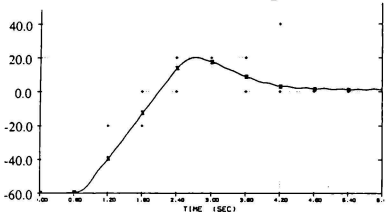


(b) Pitch Angle (Deg)

Figure 12 - Aircraft Launched, High Energy, Water Entry

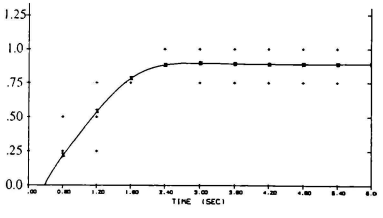


(a) Normalized Depth

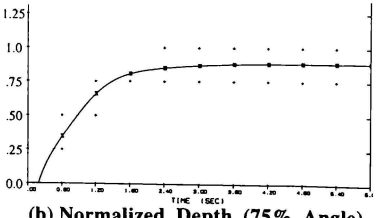


(b) Pitch Angle

Figure 13 - Aircraft Launch, Fin Activation at Water Entry

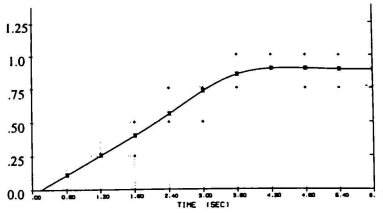


(a) Normalized Depth (50% Speed)

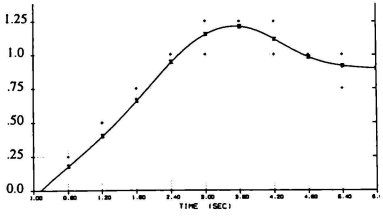


(b) Normalized Depth (75% Angle)

Figure 14 - Aircraft Launch, Reduced Entry Speed and Angle



(a)Normalized Depth, Min Entry Angle



(b)Normalized Depth, Max Entry Angle

Figure 15 - Hover Launch Pullout

THAAD Integrated System Effectiveness Simulation (TISES)

Dawn Horn (THAAD Project Office), Phil Colvert (Tec-Masters Inc.),
Becky Scrip (ELMCO)

Abstract:

TISES is a non-real-time, discrete event, engineering design-level simulation of the Objective THAAD system. The purpose of TISES is to provide the THAAD Project Office with an analysis tool to support the end-to-end performance evaluation of the objective THAAD system effectiveness in a many-on-many environment. TISES is currently undergoing modifications required to support THAAD Milestone II analysis.

By simulating the message interfaces and corresponding dynamic interdependencies between segments of the THAAD system, TISES provides the capability to support the spectrum of one-on-one and many-on-many analysis of the integrated THAAD system against TBMs under various battle conditions. TISES is being used to address critical THAAD issues. The presentation / paper will address the purpose of the simulation, types of input and outputs, and a brief summary of what is modeled within the simulation.

DISTRIBUTION STATEMENT A. Approved for public release; distribution is unlimited.

Purpose:

The purpose of TISES is to provide the THAAD Project office (TPO) with a system engineering analysis tool to support the end-to-end performance evaluation of the THAAD system effectiveness in a many-on-many environment. Results from this Government-owned simulation are being used to evaluate THAAD critical issues, verify and substantiate system requirements, and support system design decisions. The simulation models are based on existing THAAD system designs, environmental effects, and validated threats, as provided by the Government. The non-real-time version of the simulation is being used to support the Milestone decision by verifying that the objective THAAD system satisfies its performance requirements. The real-time version of the simulation is being used as a component in the THAAD Test Controller (TTC) to create a battalion-level loading environment for hardware and software in-the-loop testing as well as full theater wargaming exercises. This paper will focus on the attributes of the higher fidelity discrete event version of the simulation.

TISES is capable of simulating various system architectures, but to date has primarily simulated a THAAD battalion (BN) configuration composed of four (4) firing batteries and one headquarters battery (HHB) with a battalion TOC and two (2) additional THAAD Radars, which may be allocated among the batteries, kept in reserve, or used to provide an additional surveillance capability within the enclave. Focus has been placed on "conduct the air battle" functionality. An augmented battery configuration is also available, in which one THAAD Radar will be organic to each THAAD firing battery, with two additional radars assigned to each battalion, only one of which will have a Sensor System Interface (SSI) for remoting. Depending on the radar resources required for azimuth coverage and range, at least one additional radar may be deployed with a battery. The augmented battery will conduct operations in the same manner as a battery with one radar, except that launcher placement must be consistent with the geometric constraints of the radar. Consequently, the number of missiles available to engage targets in the field of view of a single radar may be constrained.

Application Areas:

By simulating the message interfaces and

corresponding dynamic interdependencies between segments of the THAAD system, TISES provides the capability to support the spectrum of One-On-One to Many-On-Many analysis of the integrated THAAD system against TBMs under various battle conditions. The utility of TISES is in its ability to address critical issues and provide data to support design decisions. Areas of analytical focus include:

Architecture Trades: Investigation, analysis. and high-fidelity performance assessment of defined architecture concepts as well as excursions including the assessment of integration with external systems.

Defended Area and Timeline Analysis: High-fidelity evaluation of system and element timing, element coverage, inter-element coordination, and integration issues.

Error Budget Analysis: Investigation and verification of error budget allocations across system elements and components.

Requirements Verification and Validation: Verification of allocated element requirements and aggregated system requirements.

Sensor Analysis: High-fidelity analysis of sensor suite and interaction with overall defense system including element contribution to overall system performance.

Weapon Analysis: High-fidelity analysis of integrated weapon system, including utilization of battlespace and effects on overall system performance.

Communications Architecture Analysis: Exploration of Communications Network performance, including critical timing, error tolerance, throughput, and reliability.

System Performance Assessment: high-fidelity analysis of system effectiveness and performance contributors: i.e., parametric study of various system element and parameter variations to overall system effectiveness.

Characteristics:

TISES is a non-real-time, discrete-event, engineering-design-level simulation of the THAAD system. The simulation represents the core functions necessary to execute the "conduct the air battle" functions and utilizes a Graphical User Interface (GUI) to support experiment set-up, playback, and post-experiment analysis. Within TISES, each segment model has a well-defined, generic model interface with the simulation framework. Other simulation features are described below:

TISES currently runs with the SG IRIX Release 5.3 operating system and the RATIONAL VADS Ada compiler Version 6.2.3. Turnaround times vary depending on scenario size and defense plan selected. Times can go from several minutes for a single object threat to several hours for larger scenarios. Depending on the size of the threat scenario and defense plan, TISES can require access to a somewhat large amount of physical/virtual memory. The larger threat scenarios are currently using between 200 and 250 megabytes of virtual memory.

Inputs:

TISES inputs are required in a number of data categories, which include 1) communications network connectivity and capacity characteristics, 2) threat laydown, trajectory, and signature data, 3) missile technical performance characteristics, 4) launcher technical performance characteristics, 5) radar technical performance characteristics, 6) BMC3I technical performance characteristics, 7) selectable Rules of Engagement (ROE) and Area Of Responsibility (AOR), 8) User input search and cue search commands, and 9) defense plan data.

Outputs:

Each simulation model in TISES outputs numerous records of information. These records are written in ASCII format primarily to four files: Graphics.log, Events.Log, Messages.Log. and Diagnostics.Log. The Graphics.Log and the Diagnostics.Log files are used to show a graphical playback of the simulation result and to verify the simulation worked correctly. The primary utilization of the Events.Log file and the Messages.Log file is for systems analysis. To support the user, TISES provides several post-processing analysis tools to support data reduction and review, data correlation and formatting, and statistic generation and plotting. Data reduction utilities are used to subset the logged data and

review the contents of individual datasets. In this environment, the user extracts information and constructs a predefined or customized output report. Typical outputs reports include: 1) system effectiveness, 2) system timelines, 3) radar effectiveness, 4) radar loading. 5) comm network performance, 7) BMC3I effectiveness, 8) BMC3I performance, 9) missile effectiveness, and 10) missile performance.

Simulation Functional Description:

TISES provides segment models of the THAAD Launcher, THAAD Missile, THAAD Radar, and THAAD Tactical Operations Center (TOC) integrated into a Common Simulation Framework (CFW). The CFW provides each model with support services that define how each model is integrated, synchronized, and controlled. The CFW is also capable of supporting both non-real-time discrete-event simulation or real-time distributed simulation. Multiple levels of detail are available for the THAAD Missile model and the THAAD Radar model. The high-fidelity models are used to support non-real-time execution of the simulation, whereas the lower fidelity models are used to support the real-time execution of the simulation. Low fidelity models can also be used in fast-running executions of the simulation to support quick evaluations of the defense design.

The TISES Battle Management segment model simulates the capability of the TOC to coordinate THAAD resources to provide for an effective defense. Battle Management functions represented include: Operations Management, Sensor Management, Communications, Track Management, Cueing, Classification and Typing, Defense Criterion, Threat Evaluation, Commit, Battlespace Determination, Engagement Planning and Scheduling, Fire Control, Engagement (Weapon) Control, and Kill Assessment and Re-engagement. TISES currently models multiple TOCS, each with a single collocated radar and multiple launchers. THAAD internal and some external (e.g., JTAGS) communications are modeled. The model simulates BM-to-Launcher (FOL) and BM-to-Radar (FDDI) communications links using streamlined representations of the message sets and contents. Inter-TOC communications are accomplished using a JTIDS model. A newer JTIDS model is being implemented.

Within TISES, the BMC3I Operations Management (OM) functions are performed off-line. Databases required by Engagement Operations (EO) functions are defined and populated prior to simulation execution and are provided to EO as engineering inputs. Defense design and evaluation are conducted off-line. Defended assets may be defined by the user along with their respective values and vulnerability radii. Selection may be made from a set of predefined SMPs and MPs. Areas of responsibility are specified by polygons of latitude-longitude points. Rules of engagement include factors that control the aggressiveness of the system response, the relative value of time (e.g., lost battlespace), and launcher farm load balancing. Planned features include contingency plan capabilities, a simple OSI, limited automated defense design/evaluation capabilities, missile threat origins definition, engagement subplans, and rudimentary communications feasibility evaluation.

The TISES BMC3I segment model simulates the sensor management capabilities of the TOC. Radar-loading data contained in the Resource Allocation Report message from the THAAD Radar are received and stored. The TOC accepts radar tracks but currently does not direct track resource adjustments (via the Object Tasking Command). A limited capability to direct PRI adjustments to achieve desired track accuracies within prescribed limits is included. The SSI capability includes a similar feature along with the ability to forward Resource Allocation Reports to the TOC.

The TISES BMC3I segment model supports classification and typing by accepting classification, typing. and sub-classification (discrimination) data from the THAAD Radar. Warhead typing is accomplished consistent with a priori OM guidance (default typing based on TBM type). Deconfliction is currently accomplished based on best track quality. Augmentation of the classification determination based on object metric data is planned.

The TISES BMC3I segment model currently

supports several engagement planning modes which include: asset-based, threat-based, leakage-based, and a hybrid approach. Methods of fire are not represented explicitly but are executed implicitly by means of their relative statistical merits.

The TISES BMC3I threat evaluation function is based on propagation of all potential threat tracks to impact and performing a greedy allocation of threats to assets based on the PIE, CEP, and casualty/contamination radius. Eligible threats are screened based on area of responsibility. Impact point predictions are updated whenever track accuracy improvements exceed a user-defined threshold. Overlapping damage is considered in determining damage scores. As a result, explicit threat priorities are not represented; overlapping damage considerations preclude independent threat priorities.

The TISES BMC3I segment model provides logic to perform Track Management and Cueing. Source and system tracks are correlated based on track IDs and a chi-square statistic. Track-reporting responsibilities are determined by a representation of the R2 rules (including track quality). Tracks are monitored and stale tracks deleted. External tracks (via JTAGS) are currently not correlated with THAAD system tracks. A rudimentary cued search capability (overlapping coverage regions only) to support internal radar-to-radar handover has been demonstrated. Search commands specifying volume searches may be provided as user input.

The TISES BMC3I segment model battlespace determination implements a type-dependent Pk constant over the battlespace. Battlespace is limited by, (among other factors) collocated radar field of view (including RCS and off- boresight effects), interceptor kinematic reach, a calculated (not constant) time to achieve required track accuracy, keepout altitude, and minimum flyout/seeker time. Because TISES does not perform explicit battlespace determination, holes in the battlespace as a result of seeker constraints or sun/moon/earthlimb exclusion may be determined prior to engagement planning and scheduling.

The TISES BMC3I segment model engagement planning and scheduling process models radar loading as a simple intercept rate constraint. It also considers launcher firing rates. balancing farm inventories, depleting low-inventory launchers within a farm, and the efficient utilization of battlespace consistent with the most favorable MOF. Engagements being executed by others TOCs are also factored into the determination of a near-optimal solution. A tailored maximum marginal return scheme utilizing a two-tiered objective function to effectively handle the overlapping damage problem is employed. The algorithm is capable of handling both collocated and remote radars and launchers. A look-ahead objective function based on the user selected engagement planning (EP) mode. Inventory projections are based on a meanvalue approximation.

The TISES BMC3I segment model commit functionality screens candidate pairings based on pre-commit track accuracy requirements. An adaptation of Sorrenson's equations based on some simplifying assumptions has been implemented in an effort to extend their applicability to the endoatmospheric regime. Once commit requirements are satisfied, the fire control function is performed using tactical firing tables, including all pertinent technical parameters. Azimuth dependencies are currently not considered. Consideration of altitude effects is planned.

The TISES BMC3I segment model provides logic to represent Engagement (Weapon) Control functionality. Logic is present to generate Interceptor Acquisition Commands and Interceptor Support Plans (ISP). The capability also exists to update ISPs as required. Engagement monitoring (e.g.. overdue messages) is also modeled. Planned capabilities include flyout trajectory monitoring and command destruct.

The TISES BMC3I segment model kill assessment and re-engagement functions assumed perfect lethality (i.e., hit equals kill). Re-engagement is performed as appropriate. Mult-shot (SLSn and Salvo) engagements are supported. Simple kill assessment logic based on radar and downlink data is implemented.

The TISES Communications Network model simulates the connectivity and capability of the communications system to distribute tactical

information to the individual elements deployed at various geographical locations. Different communications media are used between elements based on their particular need. JTIDS is used to connect the battle manager elements. Fiber optic links (FOL) are used to connect the launchers to the battle manager elements, and FDDI is used to connect the radar to the battle manager elements. A VR-99 model to connect the launchers to the battle manager elements in under development. The transmission media modeled include: Joint Tactical Information Distribution System (JTIDS). Fiber Optic Link (FOL), and Fiber-optic Data Distribution interface (FDDI). In addition to these, the transmission media used for the launcher-to-missile interface and the radar-to-missile interface are modeled. The specific aspects of the transmission media modeled include asynchronous message passing, connectivity constraints, link distance constraints, link throughput constraints, outgoing and incoming internal processing delays, outgoing transmission delays. and radar-to-missile communications look angle and range constraints.

The TISES medium fidelity missile segment model is a component-level simulation of the missile. This model is being utilized to support both non-real-time many-on-many runs and real-time runs of the simulation. The real-time version (within the THAAD Test Controller) was recently used to simulate the THAAD missile in Roving Sands exercises. The model includes component-simulations of the vehicle dynamics, Inertial Measurement Unit (IMU), Propulsion, Divert and Attitude Control System (DACS), Gimbaled Seeker, Transponder, and Avionics. The vehicle flyout dynamics will be simulated as an augmented 3DOF. The three translational states are updated using Runge Kutta Integration of the equations of motion assuming rigid body dynamics. The rotational states are updated using user-defined response models: instantaneous, first-order or second-order. Vehicle aerodynamic properties are simulated for boost power on flare, boost power on no flare, boost power on flare, boost power on no flare, boost power off, KV shroud. KV no shroud airframe configurations. The IMU model consists of both a three-axis gyro and accelerometers for sensing body rates and accelerations. The IMU model accounts for offset acceleration from the missile center of gravity and include error sources for scale-factor linearity, bias, noise, cross-coupling sensitivities, angular rate and acceleration sensitivities, and random walk errors. The Propulsion model simulates the THAAD booster and capabilities of the TVC nozzle system. The Propulsion model accounts for nozzle misalignments, thrust magnitude uncertainty, body frame position uncertainty, thrust provide perturbations due to atmospheric effects, varying burn time due to differing motor temperature, and fuel consumption. The DACS model accounts for fuel usage of the six attitude control and four divert control thrusters and model the constant force Pulse-Width-Modulated (PWM) type performance characteristics of each nozzle, including thruster misalignments, thrust magnitude uncertainty, thrust profile on/off delays, body frame position uncertainty, augmentation factor for jet interaction effects, and propellant usage. The Seeker model provides a functional model of the camera and three-axis gimbaled platform. The camera model simulates a single-band MWIR sensor that imposes constraints for field-of-view and sun-moon-earth obscuration. The Avionics model simulates functions that perform seeker platform control, signal processing, frame processing, threat navigation, interceptor navigation, guidance, and control of the missile. The signal-processing model provides a signal-to-noise (SNR) based detection model for point-source targets that is used to construct a list of detections from the number of true objects in the field of view. Control is provided to the user to impose or bypass CSO resolution and clustering logic. Detection errors are introduced to account for angular and radiometric precision and accuracy. Performance models are utilized to perform measurement-to-track association, warm-start track initiation, track update, discrimination, and target designation. Logic is implemented to simulate boost, midcourse. track, and endgame guidance. Navigation functionality models prelaunch alignment and uses IMU data to update the sensed missile state. The Transponder provides the capability to simulate the passing and receiving of pre-launch and in-flight messages, including

guidance updates, target updates, and target object maps. Probability draws are also included to simulate the reliability of key flyout events, including booster separation, shroud separation, message receipt, seeker activation, and target designation. A lethality assessment is made at the time of closest approach based on the endgame geometry.

A very-high-fidelity model of the THAAD Missile is currently being integrated into the simulation framework to support one-on-one analysis. The model being incorporated is the THAAD SETA STAT Missile model with modifications to allow it to interface with the TISES simulation framework for initialization data, control to execute, services to access communications, truth threat data, and data recording. The model incorporates detailed hardware models and includes a FORTRAN implementation of the missile algorithm design document (now contained in the Missile SDD).

The TISES Launcher segment model performs message and launch operations functions necessary to execute the mission as directed by the Battle Manager. In support of mission objectives, the launcher model performs prelaunch state initialization and update, launch-rate-constraint checks, launch enablement. and inventory maintenance. The launcher also provides feedback to the Battle Manager regarding the success or failure of each commanded launch.

The Launcher segment model determines if the criteria for launch is acceptable based on receipt and acknowledgment of a launch command from the Battle Manager. This determination is based on inventory counts, launch rate constraints, and sufficient time remaining to implement the launch once the command is received. If the determination results in an inability to launch, a Launch Report is returned to the Battle Manager with a failed indication and no further processing is performed. If conditions are acceptable for a launch, a missile is allocated to the engagement, and just prior to the time of ignition, the launch parameters are transmitted to the missile. The Launcher will receive a launch report from the missile and report the status back to the commanding Battle Manager.

The TISES Radar segment model simulates a multi-mode, single-face, wideband phased-array antenna with monopulse capability. The model performs resource management and surveillance, simulates measurement generation and processing, and provides engagement support functions for communications with the inflight missile. Simulated resource management is performed at the pulse-to-pulse level of detail with beam-to-beam interleaving of radar tasks. Capability for autonomous surveillance over designated search sectors and cued surveillance from handover of object tracks from BMC3I is provided. as well as self-cueing from booster tracks. Like the missile model, the TISES Radar segment model is also provided at two levels of detail.

The TISES Radar segment model pulse scheduling logic simulates the pulse-by-pulse interleaving of radar tasks. The scheduler operates on user-defined priorities, pulse repetition frequencies (PRFs), pulse limits, and parameters by task termination/transition and saturation response. Pulsewidths are selected based on the radar range equation from a list of user-defined values. Multiple pulses for each radar resource period are scheduled subject to transmit duty cycle, pulse occupancy, and range self-eclipsing constraints. Timeline deconfliction is not modeled for the transmission and reception of multiple pulses within a radar-resource period.

The TISES Radar segment model simulates the surveillance functions and capabilities for the radar in the performance of autonomous and cued searches. Each search beam is modeled spatially and temporally. A sine-space search raster is computed for each autonomous search sector. The Radar model accepts search cues and determines a hexagonal spiral scan for each one based on its cued target state and covariance. Self-cueing is modeled based on tank tracks in order to acquire separated RVs.

The TISES Radar segment model interacts with the simulation framework to generate simulated measurements for each radar pulse. On a pulse-by-pulse basis, the target returns are simulated for objects illuminated by a radar beam and used by the signal processor model to generate simulated radar measurements for each beam.

Angular Field-Of-View (FOV) limits and range sensitivity are provided for each radar site. Beam shape loss, scan loss, and atmospheric attenuation (including rain attenuation) are modeled, in addition to system losses. Video (complex) return signal voltages for monopulse sum and difference channels are calculated. Range resolution cell widths, range-Doppler coupling, and matched filter responses are modeled for different receiver bandwidths. The return voltage in each range-resolution cell in the measurement gate is sampled and voltage peaks above the detection threshold are identified. A Boolean flag activates the false-alarm model, which simulates the noise voltage spikes for a radar beam as a function of the detection threshold and the number of cells in the receiver range window. Return data are simulated, including Range, Sine Alpha. and Sine Beta (RUV) radar measurements extracted from range marking and monopulse evaluations. A user-selectable option exists to use an errored truth model for generating RUV radar measurements instead of the complex video voltage, matched filter response, and monopulse processing models.

The TISES Radar segment model simulates the processing of radar measurements by track and discrimination/classification algorithms. A 4-out-of 6 Raytheon algorithm is modeled for initiating tracks from detections. A chi-squared correlation algorithm is implemented to provide returns to track association. A seven-state Kalman filter provides updates to existing radar tracks. Typing of tracks is provided based on Raytheon's Mission Application Program (MAP) algorithm, and an associated boost/coast determination is provided. Discrimination is simulated utilizing either a truth model, a Confusion-Matrix based Discrimination Performance model, or a K-factor model. A user-selectable option exists for perfect discrimination using simulation truth data. Clustering is modeled for tracks of CSOs on parallel trajectories. A track cluster is treated as one track for the pulse scheduling purposes and requires fewer radar resources than track each object independently. Known Object Recognition (KOR) processing is implemented to purge redundant object tracks between track clusters. New tracks are split from an existing track if unassociated pulse returns accumulate within an existing track cluster. Track database limits and automatic reclustering of objects are also provided.

The TISES Radar segment model simulates the capability of the radar to provide engagement support. Radar tracking, observation, and missile uplink/downlink functions in support of planned and active engagements are simulated. High-quality tracking is performed on objects perceived as lethal to support committing a missile to engage the target. Required tracking of the missile, targets, and associated objects is planned by the radar to support the engagement timeline in the Interceptor Support Plan (ISP) message from the Battle Manager. Missile state data from the ISP is used to acquire and initiate track on a boosting missile. Guidance Data, IFTU, and TOM missile uplink messages are created by the radar from its track database and transmitted to the inflight missile in accordance with the ISP. A kill/miss decision is simulated by the radar model using an algorithm based on losing track and excess pulse returns.

The TISES JTAGS model simulates the operational and performance characteristics of the Joint Tactical Ground Station (JTAGS) as defined by the U.S. Army Air Defense Artillery School (USAADASCH). Within TISES, the JTAGS model provides a mechanism to input TMD attack warning, alerting, and cueing information into the theater of operation using theater communications networks. The JTAGS model provides the capability to receive and process space-based early warning sensor data in order to locate. count, and report ballistic missile data during the launch or boost phase. Once processed the attack warning and track report data are provided to the Battle Management elements over the Tactical Information Broadcast System (TIBS). Control over use of the JTAGS model is provided to the user through input of the operational scenario parameters. When used, the JTAGS model gives the user the capability to study and evaluate the benefit of external cueing on THAAD system performance.

The TISES Threat model simulates the trajectories of TBMs as defined in Government approved documentation by MSIC. Currently, TBMs are simulated that remain intact during the

entire flight, separated into a first-stage booster and/or a second-stage booster and a reentry vehicle. RCS values are provided from the threat model to the radar to support detection and discrimination functions. Likewise, the threat model provides radiant-intensity data for the waveband of interest to the missile model. Threat trajectory files are derived from STAMP output using a Threat Preprocessing Tool.

The TISES (Natural) Environment model includes services that provide information on the location of the sun and moon, incorporates the WGS-84 standard for an elliptical earth, and includes the U.S. Standard 1976 Atmospheric model. Effects introduced by induced environments, such as intercept debris, are planned but are not currently modeled.

The operational concept for the THAAD system design simulated in TISES is based on the timeline of messages in a single nominal engagement sequence against a single threat. TISES is capable of simulating multiple simultaneous engagements from multiple launchers at multiple threats under the control, monitoring, and coordination of multiple TOCs and supported by multiple THAAD Radars. For the purposes of illustration, the timeline has been divided into seven phases. including: Strategy, Search, Acquisition and Discrimination, Prelaunch, In-Flight. Kill Assessment, and Stop Search.

The TISES scenario beings assuming that all defense assets have been deployed to the field and the system has been initialized and is ready to conduct the air battle. During the Strategy phase, the THAAD Battalion TOC transmits the Rules of Engagement and Area of Responsibility to each of the battery TOCs. The Area of Responsibility message describes the geographical region for which the battery TOC is to assume responsibility for defending. The Rules of Engagement message directs the battery TOC to a preplanned parameter set that defines the strategy or strategies to be employed. This strategy includes the level of aggressiveness with which the battery TOC will expend resources (Radar, Launcher, and Interceptor) to negate the threat and guidance for distributing tasks over time and among available resources to facilitate increased preparedness for potential future threats. Both of these messages (along with all adjacent TOC messages within TISES) require acknowledgments confirming successful receipt. Adjacent TOC messages for which acknowledgment messages are not received are retransmitted.

The Search phase includes the generation and transmission of Search Commands, which specify search volumes and fences, and Search Cue Commands, which specify handover states and covariance. Search Cue Commands can either be scripted or generated autonomously. Autonomous generation is limited to handover of objects in overlapping coverage regions or as a cue from an external sensor through the receipt of a (JTAGS) Advisory or Surveillance message. Search Commands and Search Cue Commands, as well as all THAAD Radar task messages, require acknowledgment using the Non-Compliance Report. The report indicates the compliance or failure to comply with the command. In this setting, the Resource Allocation Report message is used periodically to report the radar resource allocation (duty/occupancy) to the battery TOC for the purpose of supporting saturation avoidance. The TISES TOC stores these data but currently does not model saturation avoidance at the level of fidelity sufficient to use the data. It is currently accomplished using an intercept rate constraint per supporting THAAD Radar.

The Acquisition and Discrimination phase encompasses the timeline starting from track initiation up to, but not including, pre-commit sensor tasking. During this phase, the tracking THAAD Radar provides threat data using the Object Report message to provide kinematic state data only and the Discrimination Report to provide typing, classification. and discrimination data in addition to kinematic data. The TOC receiving this information attempts to associate these reports with existing system tracks. creates new system track or updates existing system track data as necessary, and forwards the system track data to other adjacent TOCs using the Track Update Report when one of the following reporting responsibility conditions is met: system track initialization; new typing, classification, or discrimination data; and improved system track

kinematic data (and periodic update due). As described above. These Track Update Reports must be acknowledged by each receiving TOC. Messages not acknowledged are immediately retransmitted.

The Pre-Launch phase includes the pre-commit and prelaunch sensor and weapon tasking. The TISES THAAD Radar model currently supports a representation of the Object Tasking Command for accelerating the time at which commit quality track can be achieved. However, technical interchanges with LMMS have indicated plans to utilize Sorensen's equations as the core of pulse rate and timeline calculations for commit quality track. Unresolved design issues regarding the applicability of Sorensen's equation to the endoatmospheric intercept problem where drag uncertainties often dominate have limited the implementation of any functionality within the TISES TOC model. Instead, the TISES THAAD Radar model currently uses the option of higher default pulse rates for objects determined to be probable lethal objects.

Once the threat has been evaluated, and upon selection by the weapon assignment function of an acceptable engagement (i.e., pairing) requiring an immediate commit, the TOC assembles and transmits a Launch Command to the selected Launcher and an Interceptor Support Pan to the supporting THAAD Radar. Compliance or failure to comply with the Interceptor Support Plan is relayed back to the TOC using the Non-Compliance Report. Receipt of the Launch Command at the Launcher is communicated to the TOC via an Acknowledgment message. Insufficient interceptor inventory will result in an immediate Launch Report message back to the commanding TOC. At the same time that the Launch Command and Interceptor Support Plan are sent, Engagement/Coordination Messages are sent to adjacent TOCs to report changes in the engagement status that are not supported by Missile Launch Reports or Kill Assessment Reports (e.g.. In-flight failures, certain overdue messages).

The In-Flight phase covers the period from the time the Launch Report is transmitted back to the controlling TOC to the last messages prior to intercept. The TISES Launch Report reports the status of the launch and additionally provides an update of the available interceptor inventory at the reporting Launcher. Upon receipt of the Launch Report from the Launcher indicating a successful launch, the TOC sends a Launch Confirmation Report to the supporting THAAD Radar to initiate the execution of the Interceptor Support Plan. Independent of the launch outcome, Missile Launch Reports are sent by the engagement controlling TOC to adjacent TOCs, notifying them of the launch status.

During execution of the Interceptor Support Plan, the supporting THAAD Radar provides scheduled uplinks containing updated interceptor states, target state, and/or associated object states to the in-flight missile using the Guidance Data Uplink message, the IFTU Uplink message, and the TOM Uplink message. respectively. For short-timeline intercepts, a Guidance Update may be provided to the interceptor during boost. After boost separation, a series of In-Flight Target Updates are sent providing updates of interceptor and target states. The last uplink(s) just prior to the Target Object Map (TOM) uplink are critical to achieving divert and orient prior to acquisition of the target. The TOM is scheduled to arrive shortly after target acquisition to provide state updates for the target and any associated objects that may appear in the kill vehicle seeker FOV, thereby facilitating the interceptor's target-designation process. Upon receipt of each uplink, the interceptor responds with an immediate Interceptor Status Downlink message, which includes the onboard interceptor state estimate information in addition to the onboard estimate of the predicted intercept time. This information is forwarded through the supporting THAAD Radar back to the controlling TOC using the Interceptor Status Report message. The TOC also utilizes the interceptor's predicted time of intercept to refine the interceptor support plan schedule.

The Kill Assessment phase involves messages that occur shortly after intercept. In order to aid the kill-assessment process, the interceptor provides an Interceptor Status Downlink message, if possible, shortly after the predicted intercept time to report a probable missed intercept. In

addition, a short time after intercept, the THAAD Radar supporting the engagement transmits a Target Kill Assessment Report to the controlling TOC indicating the probable outcome of the intercept. Based on the indications in the Target Kill Assessment Report and receipt or failure to receive a post-intercept Interceptor Status Report, the controlling TOC makes a kill-assessment decision. The results of this decision are then reported to adjacent TOCs informing them of the engagement outcome using the Kill Assessment Report.

The final phase referred to as the Stop Search phase is entered when the TOC perceives the situation to stand-down the warfighting capability of the THAAD system. Currently, this phase is scripted to occur in accordance with the experiment design of the user. During this phase, a Stop Search Command from the controlling TOC is sent to the supporting THAAD Radar to terminate the execution of a Search Command. Again, as with all sensor tasks, the supporting THAAD Radar must reply with a Non-Compliance Report indicating compliance or failure to comply.

The functional capabilities simulated in TISES were derived from an in-depth review of the tactical message set defined for the DEM/VAL system and a control flow analysis of the THAAD operational concept as defined in the THAAD Technical Requirements Document. This process resulted in the identification of a streamlined message set and the development of the necessary functional capability to simulate the required operational behavior for primarily the engagement thread of the system. The message set used in TISES is currently a subset of the tactical message set defined for the DEM/VAL System. The TISES message set has been grouped into five basic categories, as described below.

The messages not modeled in TISES generally fall into one of the following categories: those required to initialize and time synchronize various components of the system, those required to report explicit health and status type data, those that communicate environment data, those that directly support man-in-the-loop functions, and those interfaces requiring algorithm designs that have not been sufficiently finalized by the THAAD Prime Contractor.

Status:

TISES development is following a phased approach resulting in multiple releases. The Initial Operational Capability (IOC) version was completed and demonstrated to the THAAD community in October 1994. A limited distribution of Release 1 was provided to the THAAD SETA and to MICOM RDEC for review and comment. Release 1 of the software did not fully represent the THAAD capabilities. Output for Release 1 was geared to software testing and product demonstration. Release 2 of the software is designed to fully represent the evolving THAAD system capabilities as design information becomes available. The primary user for Release 2 is the THAAD SETA and the Debug Team. The main focus of current activity is directed toward Release 3. Release 3 includes a one-on-one DEM/VAL configuration using the HF THAAD missile model based on the THAAD SETA's STAT model. The many-on-many release will focus on model enhancements needed to represent BMC3I Build 2.1 and 2.2 functionality. Instrumentation and improved post-processing capabilities will be provided to support the data requirements for the THAAD System Engineering/Requirements Verification (SERV) team, AMSAA, and OEC. Release 3 deliverables are being used to support analysis for the MSII DAB review.

Simulation Meets Information: Bringing Aircraft Modeling and Simulation Into the Information Age

William C. Beavin
Principal Technical Specialist
McDonnell Douglas Corporation
Saint Louis, Missouri

Abstract

Radical innovations in information and simulation technologies have been occurring at an increasing rate over the last 40 years. We are now at a point where many of the innovations in these two technologies are on a collision course. Since the 1930s, simulation has grown from providing a means to substitute for flight, to providing a substitute for an aircraft, and recently to provide a substitute for an entire system. Meanwhile, Transaction Processing Systems and Management Information Systems created the means to gather and process massive amounts of data for accounting purposes, and matured into today's distributed Decision Support Systems. These systems infused data with knowledge, producing information as a result, and thus the Information Age was born. Currently, visualization technologies are developing to allow immersion into information like never before. Client / server technologies and hyper-links are allowing distributed information access and maintenance, creating entirely new ways of interacting with information. It is through combining these emerging technologies with established simulation and information technologies that the next generation technology will emerge. No longer satisfied with simply substituting for physical entities or processing data, we find ourselves entering an era of Information Immersion and Interaction.

Linkages	Core Concepts & Components: *Reinforced*	*Overturned*
Unchanged	Incremental Innovation	Modular Innovation
Changed	Architectural Innovation	Radical Innovation

Figure 1: A Framework for Defining Innovation [1]

Background

Trends in Simulation

Looking back over the history of aircraft simulation, several milestones reflecting paradigm shifts become evident when using the relationships shown in Figure 1[1] to identify radical innovation. Innovation in both linkages and components are required to produce radical innovations, which mark significant technology breakthroughs. Some of these breakthroughs are indicated in Figure 2, with associated paradigm and technology shifts shown in Figure 3.

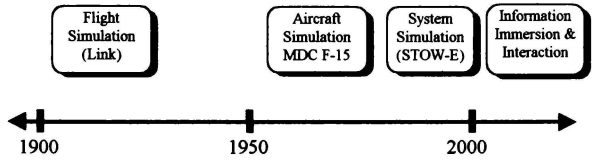


Figure 2: Some Radical Simulation Innovations

Copyright © 1997 by McDonnell Douglas Corporation. Published by the American Institute of Aeronautics and Astronautics, Inc. with permission.

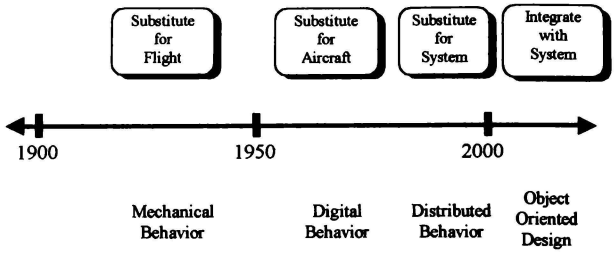


Figure 3: Simulation Paradigm and Technology Shifts

In the 1930s, the Link trainer provided a substitute for flight [12], which allowed familiarization with aircraft flight controls. The technology focus was directed towards the mechanical behavior of the aircraft.

Over the next 40 years, various improvements were made, including the integration of at first analog and later digital computers, allowing for a technology focus in simulation on the digital behavior of the aircraft. Several examples could be noted, but the McDonnell Douglas F-15 simulation, which emerged in the early 1970s, is a good example of the new simulation paradigm that had developed. Going beyond a substitute for flight, now available was a substitute for a specific aircraft, including representative flight characteristics, avionics, and weapons.

The next paradigm emerged 20 years later in the early 1990s, where distributed simulations provided a substitute for an entire theater of operations. The technology focus was on modeling and merging distributed object behaviors. A good example is the Synthetic Theater of War - Europe (STOW-E) program [8], where live and simulated forces were combined in a simulated battle at the European theater level [12].

The next paradigm steps outside the trend in simulation, which previously had largely focused technology on *behaviors*, but the current focus now stresses an object oriented integrated *design* of the simulation [13].

It can be seen that the shortening time between innovations (40 years, 20 years, and 10 years) indicates that the rate of change continues to accelerate.

Trends in Information Technology

Information Technology Systems have evolved into three basic types of systems, as shown in Figure 4 [3], consisting of Transaction Processing Systems (TPS), Management Information Systems (MIS), and Decision Support Systems (DSS).

Transaction Processing Systems are primarily concerned with capturing data at the functional level, and storing it for later use [15]. This data is typically gathered when items are bought or sold, for example, in order to better understand a customer's preferences. Another example is in manufacturing, in measuring errors versus tolerances, or failures in the system. Automated procurement systems also generate and interact with data at the transaction processing level.

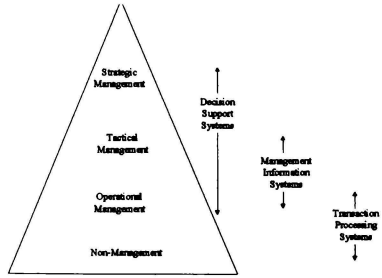


Figure 4: Information Systems Throughout an Organization [3]

Management Information Systems [16] are often used at the Operational and Tactical management levels, and are primarily used to monitor and resolve short term, event triggered issues. Staffing issues, as well as budgets and schedules are dealt with at this level with MIS tools.

Decision Support Systems [17] are used at all levels of management, and increasingly by non-management levels as well, as processing power becomes more readily available. These systems pull upon features of MIS and TPS systems, often tailored for the sort of decisions which need to be made. These decisions are often unique, require quick attention, and may not come up very often. Simplicity and conciseness in the presentation is usually required to provide all relevant information and expedite the decision.

Trends in Immersive Technologies

Current immersive technologies are already in use in experimental simulations [9]. It remains to be seen whether the continually growing abstraction between the observer and what is being observed will not hinder the fidelity of the interaction. Such a system will ultimately be useful only if the immersive interactive experience enhances understanding while still providing required functionality. This implies that a passively immersive system will be of limited value, while an actively integrated one using immersive technology could be extremely useful. Therefore, the converging technologies of information and simulation indicate that the next paradigm will go beyond simply *substituting* for physical systems, but actually *integrate* the operator within the system itself during different stages of aircraft design, development, production, and deployment.

Design

Hierarchical Layers

Object oriented design, both in simulation and information technologies, is especially focused on relationships, behaviors, and data. Of traditional importance to simulation, and increasingly for information, is also interaction. These four design elements can be shown to interrelate in layers, as indicated in Figure 5.

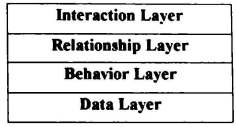


Figure 5: Object Oriented Hierarchy Layers

The Data Layer is the lowest level. It includes both data, such as engine thrust, and data access means, such as table look-ups.

Behavior builds upon the data, using the data access means provided with it. This layer describes physical properties, reactions, and processes through sequential or parallel algorithms. These behaviors exist in software objects, which correlate to physical entities.

The relationship layer describes the interconnections between the needed objects. These relationships mimic physical relationships between the entities being modeled.

The interaction layer is closest to the user, and is traditionally referred to as the presentation or user-interaction layer by the information technologist [2]. When merged with the requirements of simulation for potentially large amounts of timely user inputs in reaction to presented displays leads to the use of the term "interaction."

The order of these layers is important for several reasons, as shown in Figure 6. The layering scheme of these simulation technology components can be mapped to the information technologist's client/server domain hierarchy [19].

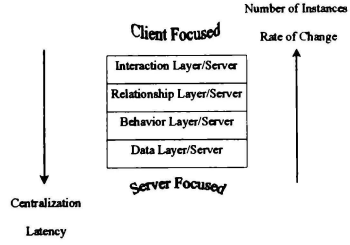


Figure 6: Hierarchical Impact

The interaction layer, being in closest proximity to the user, can be said to be user, or client, focused. In the world of aircraft simulation, this user could be an engineer at a desk developing software, or a virtual demonstration of prototype aircraft technology, or a pilot flying a full mission simulation.

The data layer, at the lowest level, is usually furthest from the typical user, but used by many. It is of primary interest to the research scientist who generates and validates data sets, such as wind tunnel data.

Effects of Hierarchical Layers

Several observations can be made of the effects of distributing these layers across a hierarchical network of servers, as shown in Figure 6. Pushing the layers away from the clients, or users, towards a common server increases the centralization of the system, making it much simpler to control and maintain, and improves the overall integrity of the system. It also simplifies software licensing and installation issues. However, centralizing towards a server increases the latency of the overall system from the client's perspective.

Pulling the layers towards the client away from centralized servers is becoming more of a viable option as memory and processing power become increasingly affordable for many users. In doing so, the client gains more control over the software being run, as well as improving the overall latency of the system. However, the number of instances of the software is also greatly increased, making control and maintenance of the system a much more complicated issue [14]. Often software is pulled towards the client for the purpose of modifying it. If multiple clients have multiple copies and they all make changes, the probability of sharing these enhancements decreases. This can be improved by allowing only one writeable copy to be checked out at a time, or making the changes on a lower level server by the process owner on that server, which can then be shared among users.

So, what does optimization mean for a system? Naturally, it depends on the intended usage and available resources. A pilot flying a full mission training simulation will need maximum fidelity and minimum latency, so local copies of software at all layers running on a dedicated computer system would be preferred in this case. An engineer at a desk developing a software module may not have the resources available for running a complete system locally, and may not desire to do so anyway to ensure baseline compatibility. This engineer really only needs the interaction layer to run locally, and probably also parts of the relationship layer, as well as any behavior or data modules which are under development by that particular engineer.

Essentially, putting simulation into the information technology paradigm, the end user is the client. This client may be a pilot in for training, a potential customer receiving a virtual demonstration of technology, or an engineer developing software on a desktop computer. The overall system will need to be flexible enough to easily optimize for all of these clients, with optimization meaning pushing or pulling interaction, relationship, behavior, and data layers either towards the client's dedicated system, or towards a centralized server.

Issues Supported by Information and Simulation Technology

Figure 7 indicates the overlapping support for key issues provided by information and simulation technologies.

Information Technology Component	Simulation Technology Component	Supported Issue
Decision Support Systems	Interaction	making decisions
Management Information Systems	Relationship	understanding relationships
	Behavior	understanding behaviors
Transaction Processing Systems	Data	gathering data

Figure 7: Issues Supported by Information and Simulation Technologies

Essentially, the simulation itself is usually used as a Decision Support System to assist in making decisions, although not traditionally described as such by simulation technologists. Information technologists have a very broad understanding of Decision Support Systems, one specific DSS tool being simulation. It is through merging the philosophies of simulation and information technologists that a much broader scope of tools becomes available to users from both technology backgrounds. The pilot using a simulation as a

training tool is rehearsing scenarios and honing skills to make appropriate decisions later. A virtual technology demonstration is intended to assist decision makers in assessing the value of a particular prototype technology. An engineer working on development of a software module must make numerous design decisions in the software development cycle, and offline integration with other modules and components is an important development step.

Understanding relationships is an area of common concern in both simulation and information technologies, especially as object oriented design emerges into the mainstream [10]. Breaking down complex systems into basic modules, and understanding the way they interact is becoming a standard method of both developing a system as well as designing software [13]. Again, information technologists have taken a somewhat broader view, and understand relationships to include the workings of design workgroups, production lines, and just-in-time delivery systems. The more specific nature of simulation can benefit from working closer with a broader perspective throughout an enterprise. Meanwhile, information technology can benefit from simulation in the time critical, tightly coupled interworkings of complex real-time simulations.

Understanding the behavior of an object is key to creating a model that represents it. Both fields have actively developed software models that exhibit the behavior of various components within systems, although the areas of focus have been different. Information technology has been focused on systems such as financial [21] and retail. Simulation technology has been focused on industries such as automotive [22] and aerospace. Therefore, not much reuse of behavior models is expected. However, the methods of creating these behavioral models using object oriented design principles and tools appears to also be converging.

Gathering data has been a primary concern for both information and simulation systems, but largely for separate purposes. Information systems known as Transaction Processing Systems have been developed to capture data from external influences, such as data describing a customer at the time of a purchase, or keep track of currency exchanges, or monitor production facility parameters. Simulation data recording has traditionally focused on recording data from within a simulation over time for later analysis to better understand the effectiveness of a process, or study a system's behavior. Again, both technologies can borrow from each other for mutual gain.

Traditional Transaction Processing Systems could be used as a tool to help provide more reliable systems by keeping track of more detailed information which can be used to study the overall design and effectiveness of the system being investigated. Simulation systems could benefit at the facility level through keeping better track of what resources are being used and at what workload. Keeping track of what specific demonstrations repeat customers have seen and who has spoken with them, and feeding this information into the customer contact manager database would assist marketing departments. Keeping track of which specific software modules were actually executed in a simulation at run-time would help measure software reuse within a facility.

Available Tools

Figure 8 lists some of the tools available which are common between information and simulation technologies. When experience with these tools, as well as others, is combined, a potentially quite powerful integrated environment becomes available. Both areas have made progress along the learning curve in integrating these components, but future progress could be enhanced through closer coordination of efforts between simulation and information technology groups.

Technology Component	Some Available Tools
Interaction	Java, VRML, HTML, Open GL
Relationship	CORBA, UML, Rational Rose, SQL, HLA, Relational Databases, Matrix-X, JMASS
Behavior	Java, VRML, HTML, CGI, CORBA, C++, Ada95
Data	SQL, DB-II, Oracle, Access

Figure 8: Available Tools

A few key items to note from the above list of tools will be illustrative of the general set of tools which are becoming increasingly available as off-the-shelf integrated toolsets.

Java is a language with several current limitations, but definitely useful to support the concepts of remote or local execution, especially in the area of graphically

oriented software at the interaction level. The key is the interpretive nature of the language, allowing the software to be written independent of the platform. One of the tradeoffs, of course, is speed. Other concerns include licensing, and distribution services which lag behind tools such as CORBA [20].

Common Object Request Broker Architecture (CORBA) appears to be a promising solution [4] to the problem described in this paper of seamlessly mapping various software services layers to either local or remote servers, depending on available resources and the intent of the particular user [5].

The Unified Modeling Language (UML) is currently emerging as a standard means of representing an object oriented design, and holds promise to assist in the object oriented design process [11].

Database tools and query languages such as SQL are quite powerful, and have matured under the use of information technologists. Simulation applications could benefit from the use of commercially available tools for managing and distributing data as has been done for the Human Genome Database at Johns Hopkins School of Medicine [6]. Storage space can be greatly reduced, and data integrity maintained, for users who send data queries to a database server, rather than copy the entire database to a local disk. The main optimization issue is the latency requirement for the data. A pilot on a simulated training mission can't wait for data queries over a network for engine performance data. However, a software engineer developing such a simulation often has a greater requirement for data integrity than reduced latency. So, database requests would optimally be handled by describing at runtime where the database queries would go - either to a local disk or memory for the full mission simulation, or a central server during the development process. CORBA systems, for example, could provide such flexibility [7] [18].

Available Databases

Many interconnected databases exist within an organization, as shown in Figure 9. Traditionally, information technologists have focused broadly on most interconnections outside of Research and

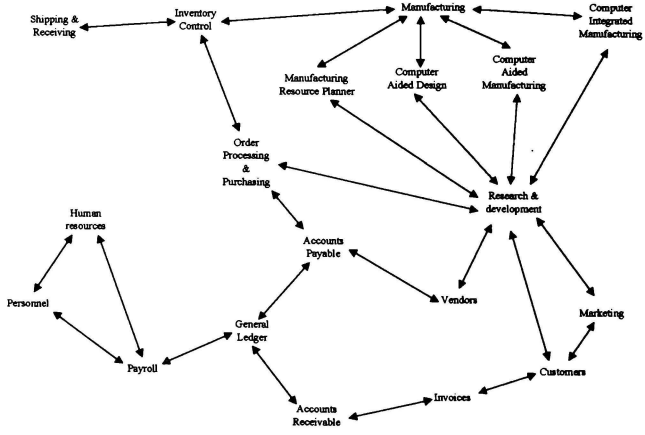


Figure 9: Databases Distributed Throughout an Organization

Development, while simulation technologists have primarily remained focused within the R&D world. Each had occasionally encroached into the other's domain until recently, when internet and intranet access has skyrocketed, and suddenly one simulation technologist can communicate with another simulation technologist through the services of an information technologist. Thus, not only are the technologies of simulation and information converging, so are the domains.

Simulation can benefit from improved links with manufacturing, purchasing, marketing, vendors, and customers, as well as internally in the R&D world.

Connecting R&D with manufacturing through resource planning, design tools, and manufacturing systems can directly enhance and streamline the design for manufacturing process. Modifications made at the CAD/CAM/CIM level can directly feed back through shared databases to R&D to analyze the modifications, and include enhancements in later designs. Including Manufacturing Resource Planners (MRP) into the R&D planning phase can provide for optimal resource allocation built directly into the design and development plan for the product.

Closer ties with purchasing through developments in electronic commerce has already streamlined the purchasing process. Closer links with vendors can improve communication of requirements and priorities.

Closer electronic ties with marketing groups can allow software to interact directly with contact manager programs to help tie customer needs to product design and development. Automatically logging customer visits and demonstrations, along with what they saw and who spoke with them, can be valuable information when maintaining customer profiles.

Increasingly, one of the most visible means of communication with the outside world, the internet web site, is fast becoming a critical interface to customers. Integration of simulation technology into the internet and intranet can provide powerful demonstrations of products and technologies to potential customers.

Improving communications within the R&D community itself is also made possible through integration of information technology. Online forms and documents are mundane, but quite efficient, and allow Transaction Processing Systems to capture form based data into MIS and DSS systems for more detailed study.

Conclusions

Both simulation and information technologies can be enhanced through closer ties and sharing of technology. A design for an aircraft modeling and simulation system which is useful throughout an organization, with access controlled at a particular level of security, for the entire life cycle of a product, must include information links with accessible computer systems throughout the organization, but especially throughout the R&D community. In order to effectively reuse software modules throughout the design, development, maintenance, and training phases of a product's lifespan, the software would benefit from a design such as that specified for CORBA compliance. Software should be designed for either local or remote execution, and so should be modular, encapsulated, compiler and platform independent, and comply with established industry standards. An integrated development approach should be established between all users, where ownership means providing access to a model and links to all supporting data and software it requires, such as an aerodynamic model package which includes the model itself, as well as links to associated data and table lookup functions.

Future Work

The primary focus of this study was determining design requirements for future aircraft modeling and simulation efforts. The next step includes the prototyping of designs and systems which begin to implement these requirements.

References

[1] Burgelman, Maidique, Wheelwright, "Strategic Management of Technology and Innovation", 1996

[2] Dickman, Alan, "Technology Strategies: Two-Tier Versus Three-Tier Apps", Information Week, November 13, 1995.

[3] Stair, Ralph M., "Principles of Information Systems: A Managerial Approach", 1996

[4] Weiss, M., Johnson, A., Kiniry, J., "Distributed Computing: Java, CORBA, and DCE", Open Software

Foundation Research Institute, http://alpha.upol.cz/java/java_rpt/corba.htm#6324, February 9, 1996

[5] Chan, F., "A report on CORBA and its Significance", http://www-cad.eecs.berkeley.edu/~fchan/doc/report_corba.html

[6] "Object Broker Server Architecture", The Genome Database, Johns Hopkins School of Medicine, http://gdb.infobiogen.fr/ob/observer.html

[7] Chan, F., "Research on OMG/CORBA", http://www-cad.eecs.berkeley.edu/~fchan/research/corba.html

[8] Keune, C.M., Coppock, D., "Synthetic Theater of War - Europe (STOW-E) Technical Analysis", August 1995

[9] Fakespace, Inc. Homepage, http://www.fakespace.com

[10] Object Management Group (OMG) Homepage, http://www.omg.org

[11] UML for Rational Rose 4.0, Rational Software Corporation, 1996

[12] Dannenberg, K., Sanders, P., "New Directions in modeling and simulation", Aerospace America, January 1997

[13] Booch, G., "Object Oriented Analysis and Design with Applications", 1994

[14] Ben-Menachem, M., "Software Configuration Management Guidebook", 1994

[15] Boockholdt, J., "Accounting Information Systems: Transaction Processing and Controls", 1995

[16] McLeod Jr., R., McLeod, R., "Management Information Systems: A Study of Computer-Based Information Systems", 1995

[17] Turban, E., "Decision Support and Expert Systems: Management Support Systems", 1995

[18] Orfali, R., Harkey, D., "Client/Server Programming with Java and CORBA", 1997

[19] Vaughn, L., "Client/Server System Design and Implementation", 1997

[20] Mowbray, T., Malveau, R., "CORBA Design Patterns", 1997

[21] Winston, W., "Simulation Modeling Using Risk", 1995

[22] Society of Automotive Engineers, "New Developments in Vehicle Dynamics, Simulation, and Suspension Systems", 1995

AIAA-97-3592

AN EVOLUTION IN ADVANCED DISTRIBUTED SIMULATION

Christina L. Bouwens
Science Applications International Corporation
3045 Technology Parkway
Orlando, Florida 32826-3299
Phone: (407) 282-6700
Fax: (407) 381-8436
Email: Christina.Bouwens@cpmx.saic.com

Anita Adams Zabek
The MITRE Corporation
1820 Dolley Madison Boulevard
McLean, Virginia 22102-3481
Phone: (703) 883-1389
Fax: (703) 883-1279
Email: anita@mitre.org

Abstract

The concept of Advanced Distributed Simulation (ADS) has evolved over the past fifteen years from the early days of SIMNET, through the Distributed Interactive Simulation (DIS) standards, the development of the Aggregate Level Simulation Protocol (ALSP), and most recently the High Level Architecture (HLA). This evolution includes the historical events, the technical architectures, and the standards to support interoperability. The historical evolution shows the architecture development paths that led to HLA today. Architectural evolution has utilized lessons learned to evolve to a more flexible architecture supporting more application domains and simulation types than ever before. Standards evolution shows a community willing to evolve its processes to meet the changing needs of the ADS community. The objective of this paper is to provide the modeling and simulation community with information on these three aspects of the ADS evolution and what it means in the years to come.

Introduction

The Department of Defense (DoD) modeling and simulation (M&S) community has been utilizing networking technology over the past fifteen plus years to create a shared synthetic environment. This synthetic environment, sometimes called Advanced Distributed Simulation (ADS), has allowed the DoD community to perform a wide variety of simulation activities which include training, mission rehearsal, after-action analysis, research and development, test and evaluation, and acquisition. As with any state-of-the-art technology, ADS has undergone evolutionary development over the years based on the experience of users and developers who wrestled with the technical problems of mixing simulation and networking technology. This evolutionary development is reflected in the historical, architectural and standards areas.

The Historical Evolution

Distributed Interactive Simulation

The development of the Defense Advanced Research Project Agency's (DARPA) SIMulator NETwork or SIMNET program during the 1980's demonstrated the capability to utilize local area network technology (later wide area networks) to link relatively large numbers of simulators. This allowed the operators of the linked simulations to participate in a common synthetic environment for the purpose of team training and mission rehearsal. In May of 1989 a symposium was conducted to explore the expansion of the concept beyond training and beyond manned ground vehicle simulations. Later that year, the first of a series of workshops to explore standards to support the interoperability of defense simulations was held in Orlando, Florida. These semi-annual workshops later became known as the Distributed Interactive Simulation (DIS) workshops.

Initially sponsored by DARPA and later by the Army Simulation, Training and Instrumentation Command (STRICOM), the University of Central Florida's Institute for Simulation and Training (IST) has been supporting the conduct of the DIS workshops since 1989. IST provided technical and administrative support to develop the first DIS draft standard which was published in June 1990. This draft became the starting point for simulation interoperability. The first draft standard addressed the format and semantics

for exchanging simulation data between simulations. The data exchanged are called Protocol Data Units (PDUs). Although document and technical support was provided by IST, the first DIS standard or "PDU" standard was really developed by an active group of industry, government and academia volunteers. Working through workshop meetings, interim meetings and email, the draft standard that was first published in 1990 became an approved IEEE standard in 1993 (IEEE 1278-1993).

By the time IEEE 1278-1993 was published, the community already found much of the information obsolete. Version 2 of the PDU standard was already underway to include support for radios, electromagnetic emissions, and simulation management. Other PDUs were being explored to support environmental data, entity control transfer and support for "live" or instrumented systems. Experience in the use of the standard protocols through various demonstrations helped to fine tune the standard and point out the need for further protocol development. One notable set of demonstrations was performed during the Interservice/Industry Training, Simulation and Education Conference (I/ITSEC) 1992 - 1995. The Warbreaker program, sponsored by DARPA, also provide valuable insight into the applicability of the DIS standard to large scale simulation problems.

While IEEE 1278 was being updated, another standard was already in development. This additional standard defined the underlying communication services required by the PDUs. A number of communication profiles are provided in this standard. As IEEE 1278-1993 version 2 went back to IEEE for balloting, the first Communications standard was also balloted through IEEE. Version 2 of the PDU standard became IEEE 1278.1-1995 and the Communications Services standard became IEEE 1278.2-1995.

IEEE approved "Recommended Practices" have recently been approved for: Exercise Management and Feedback (IEEE 1278.3) and Verification, Validation and Accreditation (IEEE 1278.4). The last DIS recommended practice represents the Fidelity Taxonomy for use with DIS (IEEE 1278.5). This draft standard is currently being balloted and should be through the balloting process by August 1997.

Aggregate Level Simulation Protocol

In the fall of 1989, about the time of the first DIS workshops, the Warrior Preparation Center (WPC) hosted ACE-89, a DARPA-sponsored experiment in distributed wargaming using aggregate-level discrete event simulations (called "constructive" simulations). While the experiment fueled user desire for such a capability, technical shortcomings revealed a need for a better approach. The MITRE Corporation was then tasked by DARPA to investigate the applicability of the basic SIMNET principles to the development of a general and systematic approach to interoperability of this class of simulation.

MITRE examined the underlying principles of SIMNET — no central node; geographic distribution; object ownership; and message-based protocol — and determined that they could be applied to constructive simulations. However, it was also determined that there are additional requirements for a general approach to interoperability of constructive simulations. These additional requirements arise not only because of the fundamental technical differences between constructive and real-time platform simulators, but also because the constructive wargames were largely pre-existing and quite disparate in implementation. These additional requirements were:

- Time management. Constructive simulations evolve time using a variety of approaches (time steps, event-driven, best-effort, and hybrids) and often in a manner that is not directly tied to wall-clock These simulation times must be coordinated so that the times for all simulations appear the same to users and event causality is maintained.
- Data management. Because each existing simulation used its own representation of data, the approach must permit simulations to share information in a commonly understood manner.
- Architecture. The approach must allow the simulations to continue to use their existing architectures.

The Interface Working Group, a consortium of academia, industry and government representing simulation developers and users, was formed in December of 1990 to evolve the ALSP concept. A prototype "confederation" of multiple copies of the WPC's ground combat simulation, GRWSIM, was developed by MITRE and demonstrated in January 1991. A follow-on prototype using two simulations, GRWSIM and the USAF's air warfare simulation, AWSIM, was demonstrated in June of 1991, and a demonstration with AWSIM and the USA's Corps Battle Simulation (CBS) that involved the actual

wargame developer (Jet Propulsion Laboratory (JPL)

These demonstrations led to the accelerated development of a confederation consisting of CBS and AWSIM for use in REFORGER 92. This development effort was funded by US Army STRICOM and DARPA and was conducted in a distributed, cooperative manner, with MITRE developing the ALSP infrastructure software (AIS) that provided common software services, JPL developing the interface and translation services for CBS, and Los Alamos National Laboratory developing the analogous software for AWSIM. The confederation had its first user test in March of 1992, with a follow-on test in May 1992. As a risk-reduction measure for REFORGER, the confederation was first used to support US Army V Corps exercise CENTRAL FORTRESS in July 1992 and US Forces Korea exercise Ulchi Focus Lens in August.

The success of these exercises, and ultimately REFORGER in October 1992, established ALSP as the mechanism for interoperation of constructive wargames, and this particular confederation, subsequently named the Joint Training Confederation (JTC), as the simulation tool of choice for many of the major US training simulation centers. The evolution of the JTC, and concomitant evolution of the AIS and supporting protocols, has continued under the auspices of the IWG, with a new confederation fielded every year. The 1997 JTC consists of 12 simulations representing all four services as well as functional specialties such as logistics, intelligence, electronic warfare. The 1998 JTC is the last planned enhancement. Other confederations of constructive simulations have also been developed using ALSP by the SHAPE Technical Centre (now NC3A), the US Army at Fort Leavenworth and Huntsville, and the US Air Force at Hanscom Field.

for CBS) was conducted in September 1991.

High Level Architecture

In 1990, the Defense Modeling and Simulation Initiative sought to bring together DoD simulation technologies to support a wide variety of modeling and simulation (M&S) needs which extended beyond the initial training application to research and development, acquisition, test and evaluation, concept demonstration, and other domains. To support this activity, the Defense Modeling and Simulation Office (DMSO) was formed to be the DoD lead for M&S development and support.

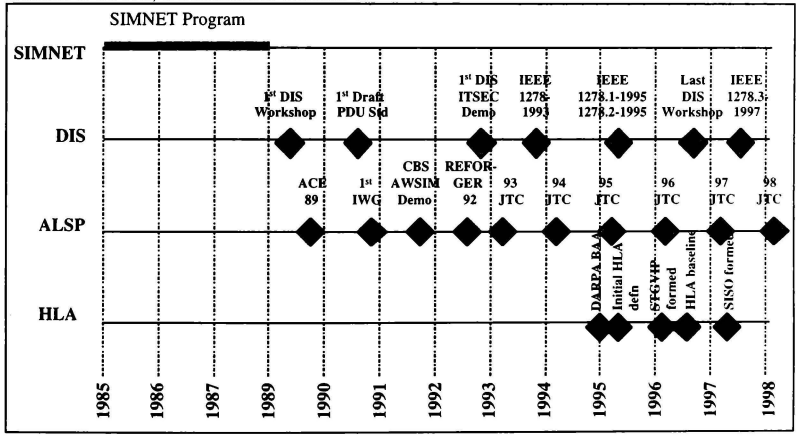


Figure 1 - The Historical Evolution

With the improvements in technology and lessons learned in both the DIS and ALSP communities, DoD set out to develop a unified ADS architecture that would provide a framework for all M&S interoperability. Beginning with three contractor developed ADS architecture proposals, the High Level Architecture (HLA) was developed. In 1996, numerous prototyping activities were sponsored to demonstrate HLA's utility and to "wring out" the architecture.

The success of these prototyping efforts called, "protofederation experiments," led to the adoption by the DoD of HLA for all future funded M&S efforts. Since 1996, DMSO's efforts have focused on evolving the HLA by extending and exploring its capabilities. Several efforts are on-going which are working to provide a migration path for existing DIS-based systems to become HLA compliant.

A summary of the history of DIS, ALSP and HLA is shown in Figure 1.

The Architectural Evolution: DIS, ALSP and HLA

DIS Architecture

The DIS community never completed development of a formalized, documented architectural concept although many proposals were discussed and presented at the DIS workshops. Experience with the existing DIS components also revealed some limitations of the current approach and the need to migrate to a more flexible, advanced approach; one which could take advantage of supporting technology developments in networking and object-oriented products. DIS needed an architectural framework and a strategy for developing the next generation DIS protocols. The HLA provided that architectural framework.

Using a network architecture perspective (Figure 2), DIS consists of a number of components. The Simulation Application represents the simulation or model that is participating in a DIS exercise or experiment. The Middleware piece is additional software created to allow the simulation application to communicate with the rest of the world using DIS. Functions such as dead reckoning, data transformation (and creation), and processing of incoming entity data are just a few of the functions performed here.

IEEE 1278.1-1995 defines the PDUs or messages along with the semantics of their use. Here the PDUs are assembled as specified by the standard. The PDUs would then be forwarded to the component which transports the PDUs to the proper network location(s). This component includes UDP or TCP, IP, and the appropriate link layer protocol based on the type of physical network used. This component is defined in IEEE 1278.2-1995.

The underlying physical network provides the physical transport of the PDU information to other hardware systems on the network.

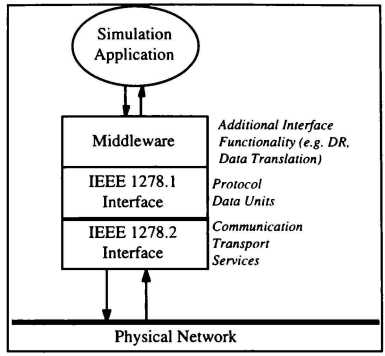


Figure 2 - DIS Components

Other IEEE documents support the two IEEE standards utilized here. IEEE 1278.3, IEEE 1278.4 and IEEE 1278.5 are supporting Recommended Practices which provide guidelines and help to define the processes associated with the application of the DIS standards.

ALSP Architecture

To generalize the approach and to meet the specific requirements of constructive simulations, ALSP departed from both SIMNET and DIS in that it provided common software services, ultimately in the form of the ALSP Infrastructure Software (AIS). The ALSP architecture is depicted in Figure 3.

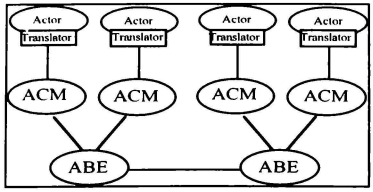


Figure 3 - ALSP Architecture

The AIS is composed of one copy of the ALSP Common Module (ACM) per simulation, or actor, participating in a confederation, and one or more ALSP Broadcast Emulator (ABE) per geographic site. The ACM provides time and data management services and the ABE provides data distribution services. In addition, ALSP Control Terminals and ALSP Confederation Management Tools can be attached to the confederation to provide the confederation operator with monitoring and control capabilities. The AIS is analogous to the HLA RTI; the intersection of the services provided by the AIS and the HLA RTI is significant although there are services that are uniquely provided by each.

The actors are modified to interact with the AIS via a translator. The translator accommodates two types of ALSP messages: actor-ACM and actor-actor. Actor-ACM messages are defined in the ALSP Technical Specification and are used by the actor to invoke and transact with services provided by the AIS. The actor-ACM messages are analogous to the RTI Specification.

Actor-actor messages support interactions between battlespace objects in a confederation. These messages are independent of the AIS and are specific to a specific confederation. The set of actor-actor messages used in the JTC is documented in the JTC Operational Specification. The set of actor-actor messages used by a confederation is analogous to an HLA Federation Object Model.

HLA

The HLA is defined by: The Rules, the Interface Specification and the Object Model Template. The HLA Rules are "a set of rules which must be followed to achieve proper interaction of simulations in a federation. These describe the responsibilities of simulations and of the runtime infrastructure in HLA federations."[3] The Interface Specification provides the "definition of the interface functions between the runtime infrastructure and the simulations subject to the HLA."[3] The Object Model Template provides "the prescribed common method for recording the information contained in the required HLA Object Model for each federation and simulation."[3]

The functional view of the HLA architecture (Figure 4) shows the components of a "federation." Simulations, support utilities and interfaces to live players are all interconnected using a single runtime infrastructure (RTI). The RTI is responsible for providing a number of services to each "federate" which interacts with the RTI. Unlike DIS, the network services are "hidden" from the federates, as the RTI provides this functionality. The Federation Object Model (FOM) is defined using the guidelines provided in the OMT. The FOM provides the definition for the types of objects and interactions that will be supported by a particular federation. The Interface Specification defines how federates interact with the RTI, allowing the federates to utilized the services offered by the RTI.

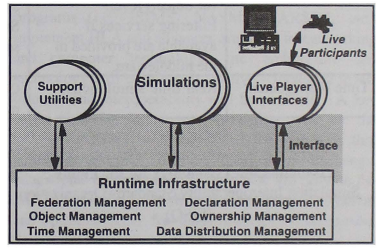


Figure 4 - HLA Functional Architecture

Table 1 compares general characteristics of the three architectures.

Table 1 - Architecture Comparison

	DIS	ALSP	HLA
Information and semantics for exchange	Generally defined by the PDU standard. More specifically defined through meetings and general agreement among exercise/ experiment participants.	Specific to a confederation. No standard format. Defined for the JTC in the JTC Operational Specification with more specific information found in the various interface control documents.	Defined in the Simulation and Federation Object Models. General and specific information is found in the FOM for a particular federation
Network access/interface	Provided by a network interface which utilized protocols defined by IEEE 1278.2. User developing the interface needs to provide access to the network or utilize commercial interface software such as VR-Link™[8]	Provided by the ALSP Infrastructure Software (AIS). The AIS transports data on behalf of an "actor", selecting the network interface required. The actor needs no knowledge of the network.	Provided by the HLA RTI. The RTI transports data on behalf of a federate, selecting the proper network services as defined in the FOM. The Federate needs no knowledge of the network.
Data Management	Ownership at the entity (or object) level. Filtering services if available are provided in the middleware.	Ownership at the object attribute level. Subscription and filtering services.	Ownership at the object attribute level. Subscription and filtering services.
Time Management	Real time simulations.	Conservative time management scheme.	Multiple time management services.

Promoting the Technology for the Future - Simulation Interoperability Standards Organization (SISO)

Re-Engineering the Standards Process: The Vision Implementation Plan

On February 13, 1996, the Distributed Interactive Simulation (DIS) Steering Committee committed to the following statement: "The DIS Steering Committee desires to support the entire DoD Modeling and Simulation community with a set of standards which include the High Level Architecture, and will embark on whatever organizational and management restructuring is required to accomplish this."

With this as direction, the Special Task Group: Vision Implementation Plan (STGVIP) was formed, with representatives from the DIS Steering Committee and key M&S communities, to examine the changes in both operations and organization required to make this commitment a reality.

The STGVIP produced the Vision Implementation Plan in support of four goals. The first goal was to establish a requirements-based, top-down process for the timely development of IEEE and non-IEEE products. The second goal was to separate the forums and management associated with IEEE standards development activities and conference style technical interchange activities. The third goal was to provide for the transition and orderly completion of current DIS products. The forth goal was to provide an open forum for commercial M&S developers and component vendors to demonstrate their products."[1]

The Vision Implementation Plan was approved by the DIS Steering Committee in August of 1996 – the result of the implementation of the plan is the Simulation Interoperability Standards Organization (SISO).

SISO Overview

Following the Vision Implementation Plan, the last DIS workshop was held in September 1996 and SISO was born at the 1997 Spring Interoperability Workshop (SIW) in March 1997. SISO seeks to facilitate simulation interoperability and component reuse across the DoD, other government, and non-government applications. SISO seeks to provide a forum for the interchange of new ideas, concepts, and technology across the broad M&S community; to disseminate these ideas; to educate M&S practitioners and sponsors regarding their implementation; and to support the development of standards, practices, and guides for use in various applications."

The organization and operation of SISO are governed by several high level operating principles designed to ensure the organization conducts its business in a manner that encourages: organization, responsiveness, responsibility, quality, discipline, fairness, openness and consensus. In standards development, these operating principles also include generality, stability and supportability.

The Organization

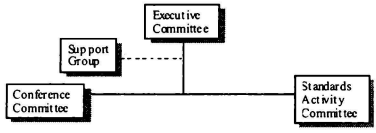


Figure 5 - Simulation Interoperability Standards Organization

The organization structure of SISO consists of three interacting operating committees (Figure 5).

- The Conference Committee (CC) provides oversight for the SISO conference activity and has the primary responsibility of organizing the semi-annual Simulation Interoperability Workshops (SIW). These workshops sponsor a number of Conference Forums which provide a mechanism for exchanging information and new ideas from within various components of the Modeling and Simulation community. The Conference Forums also help identify the needs and candidates for standardization, handled through the Standards Activity Committee.
- The Standards Activity Committee (SAC) provides oversight for the SISO standards development activity and has the primary responsibility of developing the standards and related products to support interoperability in the M&S community. The SAC conducts its development activities through a number of Standards Development Groups which operate as strongly focused, task organized groups concentrating on the development of standards and their related products.
- Both the CC and SAC are under the direction of an Executive Committee (EXCOM), which provides overall governance for SISO.
- The Support Group is responsible for providing document development and distribution support, conference management, and electronic information support (ftp, email, www, etc.).

Evolution for the Future

The historical evolution of ADS provides the basis for where it is today and begins to lay out the path for ADS evolution in the future. This evolution currently finds it basis in the HLA. As existing programs (i.e. JSIMS, NASM, WARSIM, etc.) implement HLA and advances are made in networking and computer technology, the architecture and technology will continue to evolve. Increased levels of reuse and interoperability will be possible. A key component for future interoperability will be the success of SISO workshops and the standards that SISO develops. SISO information exchange, educational components and standards will provide the ADS community with the tools needed to implement and optimize ADS for their simulations supporting their unique domain areas.

The future of ADS looks bright both for the DoD community as well as the non-DoD community. Flexible architectures supported by openly developed standards will facilitate interoperability and provide the M&S community with a starting point for supporting shared, virtual environments.

Acknowledgments

The authors would like to acknowledge the support of SAIC and The MITRE Corporation for the development and presentation of this paper.

About the Authors

Christina L. Bouwens

Christina Bouwens is a Senior Simulation Systems Engineer for SAIC. Ms. Bouwens is part of SAIC Orlando's Research Team and is currently the Project Director on the DMSO/STRICOM sponsored HLA Support Experiments project. Ms. Bouwens has been an active part of the interoperability standards development community since 1989. She has participated in DIS standards development since its inception in late 1989, helping to draft and gain approval for the first DIS PDU standard and chairing the development of the IEEE 1278.2-1995 Communications Standard. Ms. Bouwens was a member of the DIS Steering Committee from 1992 - 1995 while chairing several DIS working groups. She chaired the STGVIP which developed the Vision Implementation Plan. In addition to her DIS activities, Ms. Bouwens actively participated in the ALSP IWGs from 1992 - 1995. Currently, Ms. Bouwens is a member of the SISO Transition Team and was recently elected to SISO's first EXCOM. Ms. Bouwens holds an M.S. degree in Mathematical Science from the University of Central Florida and a B.S. in Mathematics from Geneva College.

Anita Adams Zabek

Anita Zabek is Program Area Manager in the Advanced Technology and Information Systems Division at MITRE for ADS programs including STOW (DIS), ALSP and the HLA. As project leader for CBS ALSP work at the California Institute of Technology's Jet Propulsion Laboratory, Ms Zabek participated in early ALSP prototypes and the first fielded JTC beginning in 1991. Ms Zabek later joined the MITRE Corporation and led the MITRE ALSP Systems Engineering activity and chaired the IWG. Since 1996 Ms Zabek has supported the JSIMS Joint Program Office in the development of the replacement to the JTC. Ms Zabek participated in the STGVIP, the SISO Transition Team, and has co-chaired the Interim SAC. Ms Zabek holds an ME and a BS in Systems Engineering from the University of Virginia.

References

Seidel, David, ALSP Program Status and History, The MITRE Corporation, March 1993.

Aggregate Level Simulation Protocol Technical Specification, ALSP Infrastructure Software, Version 7.4, The MITRE Corporation, August 1996.

1997 ALSP Joint Training Confederation Operational Specification, The MITRE Corporation (http://alsp.ie.org/alspguest/documents/op_specs/.html/JTCopspecs.html)

IEEE Standard for Distributed Interactive Simulation - Application Protocols, IEEE 1278.1-1995, IEEE, April, 1996.

IEEE Standard for Distributed Interactive Simulation - Communication Services and Profiles, IEEE 1278.2-1995, IEEE, April, 1996.

Special Task Group: Vision Implementation Plan, Vision Implementation Plan, DIS Steering Committee, August 1996.

SISO Policies and Procedures Document (DRAFT), SISO Transition Team, June 1997.

**SISO and DIS related documents can be found at: http://siso.sc.ist.ucf.edu

---

* "HLA Overview", HLA Tutorial, 1997 Spring Simulation Interoperability Workshop, March 1997.

† Ibid.

‡ Ibid.

§ VR-Link is a trade mark of MaK Technologies.

¶ Special Task Group: Vision Implementation Plan, "DIS++ Vision Implementation Plan," August 1996.

SISO Transition Team, "SISO Policies and Procedures Document - DRAFT", June 1997.

AIAA-97-3593

TRANSITIONING FROM DIS TO THE HIGH LEVEL ARCHITECTURE (HLA)

Deborah Fullford

MäK Technologies
185 Alewife Brook Parkway
Cambridge, MA 01238, U.S.A.

ABSTRACT

In the mid 1980's the U.S. Department of Defense's, Defense Advanced Research Projects Agency (DARPA) launched the SIMNET project, giving birth to a technology era and networking protocol presently known as Distributed Interactive Simulation (DIS). The DIS Protocol is a widely used IEEE Standard. At first it was primarily used by the military for mission rehearsal, training, and weapons evaluation. Today, other areas, such as transportation, medical care, entertainment, Internet commerce, and manufacturing, are using DIS-compliant simulations to meet their needs.

Less than a decade after its introduction, DIS is maturing into a new generation of software technology known as the High Level Architecture (HLA). HLA was developed by the Defense Modeling and Simulation Office (DMSO) as part of their goal to provide a common technical framework for the entire modeling and simulation community. The HLA was designed to increase interoperability and code re-use of modeling and simulation components. The next generation of DIS, will be based on HLA.

This paper will provide an overview of DIS and HLA and will discuss the necessary transition. First, the fundamental concepts of DIS will be discussed. They include peer-to-peer connectivity, protocol data units, and dead reckoning. Next, we will provide an introduction to the fundamental concepts behind HLA. In particular, an overview of both the Object Model Template (OMT) and Runtime Infrastructure (RTI) will be provided. Finally, we will discuss ways to transition DIS simulations to HLA.

INTRODUCTION

Over the past 30 years, the military has developed computer-generated synthetic environments to train people in very complex, dangerous, or financially risky tasks. However, since the technology for developing synthetic environments was very expensive, simulators were only developed for systems in which the cost of failed performance outweighed the cost of the training simulator.

The solution of this problem led to the invention of a technology called Distributed Interactive Simulation (DIS), a networking technique for connecting thousands of simulators together on a computer network. Instead of spending $50 Million for a single flight simulator, the demand was created for thousands of simulators to be networked in a synthetic environment for training groups of combatants in collaborative tasks.

Less than a decade after its introduction, DIS is maturing into a new generation of software technology, known as the High Level Architecture (HLA). The following paper will provide a technical roadmap of the transition from SIMNET, to DIS, and finally to HLA. Historical as well as technical information will be provided.

HISTORY OF SYNTHETIC ENVIRONMENTS IN THE MILITARY

Until about 1983, the common perception of a training simulator in the Department of Defense was that of a stand-alone, high-fidelity device designed to train a single individual, or single crew, in the skills necessary to operate a weapon system; skills such as flying, shooting, targeting, etc. The goals of the engineers who designed and built these devices was to provide the highest possible degree of realism to the crew, accurately emulating the real-world cues that the crew would experience in an actual combat situation. These lofty goals tended to drive the performance requirements and prices of these systems through the roof.

Depending on performance capabilities, these simulators could cost anywhere from $1 Million to $50 Million each. Very often, the simulator would be more expensive than the vehicle it simulated. The cost per man-hour of training time was staggering because each simulator accommodated only a single crew (often only

"Copyright © 1997 by Deobrah Fullford. Published by the American Institute of Aeronautics and Astronautics, Inc. with permission."

one person), while the training facility required several operators to run the simulation. If a $50 Million simulator were operated for 8 hours a day, 200 days per year, for 10 years, it would cost $3,125.00 per hour to amortize its purchase price, without considering hourly operating costs or repairs.

Moreover, while these high-fidelity simulations could be used for individual crew training, they could not address the need to train teams of soldiers in collective or collaborative tasks such as practice of tactics, strategies and procedures. To train these groups, it was necessary to transport all participating soldiers and equipment to an exercise site to rehearse battle scenarios. These live training exercises, often held in unpleasant and hard-to-reach locations such as the North Pole or the Sudan Desert, usually concluded with a small percentage of casualties, and even some fatalities.

Scheduling such a live exercise can be as daunting a logistical nightmare as scheduling an actual war, and may take several months to complete. This limits the number of collaborative training exercises to one or two per year.

Beyond their cost and logistics problems, live exercises are a security nightmare. To provide a realistic training environment, the exercises must be carried out in an area which closely resembles the area of conflict. These areas may be close to enemy land where an exercise may be easily observed, or on U.S. soil where the exercise may be observed via foreign satellites.

With the live training exercises' associated loss of life, high costs, poor security, and infrequent staging, the need for a better collaborative training solution was obvious. In 1983, Air Force Lieutenant Colonel Jack Thorpe, of the Defense Advanced Research Projects Agency (DARPA), decided to solve this problem.

The specific mission of DARPA was to fund high risk/high payoff advanced technologies which could be used by our armed forces to maintain an advantage on the battlefield. Almost all exotic technologies in the US arsenal, including Stealth radar avoidance, cruise missiles, laser guided weapons, Virtual Reality, and even the Internet, have all come from DARPA projects.

Jack Thorpe, along with his consultant, Colonel (retired) Gary Bloedorn, initiated one of the most influential technology revolutions ever to have occurred within the DoD. Jack and Gary envisioned a multi-user computer-generated synthetic environment where collaborative training, weapon concept development and tactics development could be performed in a very controlled, cost effective, safe, and secure manner. The outcome was SIMNET (SIMulator NETworking), a major paradigm shift for the training community. Instead of developing a small number of high-fidelity, multimillion dollar simulators for part-task training, DARPA developed inexpensive $250,000 simulators which had less fidelity than their older cousins.

"Selective fidelity" was a buzzword coined in the SIMNET program, meaning that a simulator need only be good enough to accomplish the training goal at hand. Hundreds of these "selective fidelity" simulators were networked together to provide an electronic world for collaborative tactics development and rehearsal. The tank simulators which SIMNET fielded had relatively low resolution (320 X 128 pixels per vision block), a relatively low frame rate (15 Hz), no motion base, an inexpensive fiberglass hull, and a reduced set of controls. For an 8 hour a day, 200 day a year, 10 year service life, the amortized cost of a SIMNET simulator costs only $15.63 per hour.

Proponents of high-fidelity simulators and live training exercises vilified the SIMNET system, claiming that it's cartoonish graphics and reduced fidelity man-machine interface rendered it useless as a training device. However, for its targeted intention, namely collaborative training, the SIMNET system ultimately proved to be an excellent value. Over its 10 year life span (now entering its twilight) SIMNET enjoyed widespread success and acclaim for its reduction of training costs, increased quality and quantity of tactical team training, and usefulness as a testbed for new weapon concepts.

SIMNET's biggest accomplishment, however, was the spawning of a standard networking protocol which enables heterogeneous simulations to interact in a shared synthetic environment. This standard, called the DIS Protocol, has forever changed the way the Pentagon trains our soldiers. DIS compliance is now mandatory for all new training devices procured by the DoD.

There is still decades of research and development to be done in the areas of networked simulation for collaborative training. In September 1996, Dr. Paul Kaminiski, the Under Secretary of Defense for Research and Acquisition, sent out a memorandum stated that all future simulation be based on HLA. As a result, DIS, like other technologies, is adapting to fit into the HLA paradigm.

OVERVIEW OF DIS

Technical Details

The DIS Protocol consists of some 29 different network packets, known as Protocol Data Units (PDUs) which

are passed among simulation nodes. The structure of the PDUs are outlined in a DIS standards document. The most current document being the "IEEE Standard for Distributed Interactive Simulation - Application Protocols" which outlines the IEEE 1278.1 DIS Standard PDUs (IEEE 1995).

The application specific information is encapsulated in UDP/IP Ethernet frames and transported in broadcast mode (one to all). The packets can be sent over any network media, from voice grade phone lines to ATM switches. More bandwidth allows a greater number of entities to be supported. DIS has also been used over the Internet using IP multicasting with excellent results.

A DIS PDU can describe the state information of a dynamic entity (Entity State), a combat event (Fire and Detonate), a resupply interaction (Logistic), or an electromagnetic emission such as light, radar, or energy weapon emissions (Electromagnetic Emission). Because the heritage of the protocol is military in nature, versions up to DIS 1278.1 have retained many military specific fields in the packets. However, current proposals for the next generation protocols remove most of the military content and provide a much more efficient, reconfigurable, and general purpose packet structure.

DIS uses predictive algorithms, known as dead reckoning, that allow entities on the network to greatly reduce the frequency of rebroadcasts of state. Each entity broadcasts its type, location, velocity, acceleration, orientation, and angular velocity. All the receiving simulators can then propagate the sending entity into the future, relieving the sending entity of the responsibility to continually rebroadcast. When the error between the exact position of the entity and the predicted position exceeds a certain threshold, the sending entity will update the network with its new kinematic state.

Each entity sends out information on how it should be dead reckoned. The Dead Reckoning Models (DRM) notation consists of three elements:

DRM (F or R, P or V, W or B)

The first element specifies whether the coordinate system of the entity is fixed (F) or rotating (R). The second element specifies whether rate is constant (P) or varying (V). The third elements specifies the coordinate system to be used, either world (W) or body (B). (IEEE 1995). A common dead reckoning algorithm used is DRM (F,P,W), shown below in Equation 1.

$$P = Vo + Vdt \quad (1)$$

Networked latency and dropped packets can cause a jittering effect when visualizing the location of the entities. This occurs when there is a discrepancy between the values most recently received from a PDU and the calculated dead reckoned values. To counteract this, several visualization tools use translational and rotational smoothing to interpolate between the dead reckoned location and the location received from the PDU. As a cautionary note, high level smoothing (such as rotational) utilizes precious CPU resources.

DIS enables disparate simulators to interact in a shared virtual world. In order to ensure interoperability between these simulations a common coordinate system is used to transmit the entities' state within the shared synthetic environment. Each simulator needs to be able to transform between their local system and the DIS coordinate system, knows as the Geocentric Cartesian Coordinate System (GCS). It is a right-handed system with the origin at the centroid of the earth. The positive X-axis passes through the Prime Meridian at the equator, the positive Y-axis passes through 90 degrees East Longitude at the equator and the positive Z-axis passes through the North Pole (IEEE 1995).

The DIS Entity State packet, which makes up most of the network traffic in DIS 2.0.4, is about 140 bytes long, and is broadcast anywhere from once every 30 seconds, to four or five times per second. The next generation will further increase efficiency, using a trimmed-down Entity State packet that reduces network traffic by a factor of 3 or better (see Figure 1) (Taylor 1996b).

Benefits

The DIS architecture provides flexible tradeoffs between computational loading, positional error, and network bandwidth. If highly accurate position is required, the error threshold can be reduced, which will result in more network broadcasts of state. Conversely, if the only network bandwidth available is that provided by a 9600 baud modem, and 25 entities are required on the system, the error thresholds can be increased, and more compute-intensive prediction algorithms can be used on the receiving ends. State-of-the-art DIS systems currently in use by the government are running over 10,000 entities on Ethernets, long haul networked from several locations. Future developments, such as low-cost commercial data services (bi-directional cable TV, ISDN, etc.), next generation protocols, and support for multicast routing, will enable DIS exercises to reach the 100,000 player level.

Another important feature of DIS is its self-healing nature. When a new entity enters the world, he begins

to broadcast Entity State packets. If recipients have never heard from this entity before, they simply add him to their remote entity database. If an entity is not heard from within five seconds, recipients will time him out, removing him from their remote entity database. Players can enter and leave at will without disturbing other participants, causing no effect other than their appearance and disappearance. Dropped packets don't cause system failure. In this architecture there is no central server, thus no single point of failure.

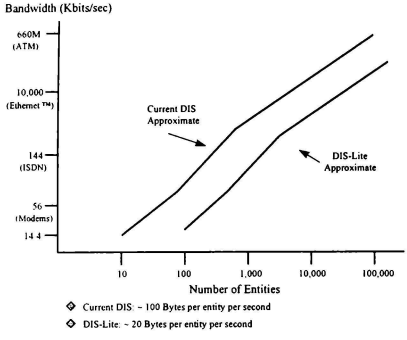


Figure 1: Bandwidth Requirements for Current and Future DIS Versions.

Heterogeneous nodes can interact with each other using DIS. The network protocol provides a standard mechanism for communication between simulators which may have radically different architectures. An entity broadcasting an Entity State packet is simply informing the network of what kind of entity it is, and its kinematic information. Two different recipients of the same Entity State packet may render the remote entity with very different levels of fidelity. For example, a PC may render a remote F-15 with only 10 flat shaded polygons, and may only have an internal simulation frame rate of 5 frames per second. A high-end Silicon Graphics workstation may render the same F-15 with 500 Gouroud shaded, phototextured polygons at a rate of 60 frames per second. In the entertainment world, this would allow a high-fidelity arcade-based system costing tens of thousands of dollars to interact, via the Internet, with a $200 home-based game system.

As compared to the computational power necessary to render out-the-window images in real-time, the DIS prediction algorithms do not represent a significant load. The most expensive prediction algorithm that DIS commonly uses consumes about 100 floating point operations per remote entity per simulation frame. Considering that the average PC can do about 10 million of these operations per second, and a simulation might run at 20 frames per second, a PC can handle several thousand remote entities using the most expensive algorithm. Naturally, this load must be balanced with other computational tasks, but this is a good indication as to the level of hardware necessary to accommodate fairly large DIS entity populations in a virtual world.

Each year, a demonstration of DIS interoperability is conducted at the annual Interservice/Industry Training Systems and Education Conference (I/ITSEC). Booths representing over 50 different organizations are networked together for a simulated exercise. DIS node systems, ranging from PCs to high fidelity flight simulators with multi-million dollar image generators, are connected for a week of interactive exercises. This is the third straight year such a demo was shown.

Though there are several commercial-off-the-shelf DIS developer's toolkits available, most of the industry uses a product called VR-Link™ produced by MäK Technologies of Cambridge, MA (www.mak.com). VR-Link, developed by some of the original SIMNET engineers, currently dominates the networked simulation market worldwide.

OVERVIEW OF HLA

Introduction to HLA

Although the existing DIS protocols work extremely well for certain applications, the DoD needs increased interoperability among simulations for which the DIS protocols may not be well suited, for example, higher order war game models. This lead the DoD's Defense Modeling and Simulation Office (DMSO) to establish a common technical framework to facilitate both the interoperability between the wide spectrum of modeling and simulation applications and the reuse of the modeling and simulation components. This common technical framework includes the HLA and is considered the highest priority effort within the DoD modeling and simulation community.

The HLA defines a set of rules governing how simulations, now referred to as federates, interact with one another. The federates communicate via a data distribution mechanism called the Runtime Infrastructure (RTI) and use an Object Model Template

(OMT) which describes the format of the data. The HLA does not specify what constitutes an object (objects are the physical things that are going to be simulated, such as tanks and missiles), nor the rules of how objects interact. This is a key difference between DIS and the HLA.

The RTI lets different types of systems interact. These systems can include simulations which run faster than real-time and simulate objects which are hierarchical aggregates of individual entities (platoons, companies, or battalions) all the way to high-fidelity engineering models which run much slower than real time and simulate individual subsystems with very high accuracy. In the DIS paradigm these two applications would not be able to interact.

However, the strength and flexibility of HLA is also its weakness—unless all the HLA simulators in an exercise agree on a single Federate Object Model (FOM) they will not be able to interoperate even though they are HLA compliant. The FOM describes the objects and interactions involved in the federation execution.

Besides facilitating interoperability between simulations, the HLA provides the federates a more flexible simulation framework. Unlike DIS where all simulations receive every piece of data broadcast, the federates now have the ability to specify:

- What information they will be producing
- What information they would like to receive
- The data's transportation service, e.g. reliable, best effort
- Whether or not the federation's timing mechanism is synchronous or asynchronous.

The above points make it possible to have more simulations on a network at one time because the amount of data being sent is reduced. The simulation software is also simplified because it does not need to process extraneous information.

Technical Details

The two major components of HLA are the Runtime Infrastructure (RTI) and the Object Model Template (OMT). The OMT is used to describe the data and metadata that is transmitted between federates. The RTI is a software application which handles the distribution of data.

Object Model Template

The Object Model Template (OMT) is used to define simulation object models (SOMs) and federate object models (FOMs). The FOM is used by the participating federates to specify how information is sent. The OMT provides a standard framework for describing a simulation in terms of its objects and the interactions between objects (DoD 1996b).

The OMT provides a standard format for describing a simulation in terms of its objects and the interaction between objects. Again, objects are the physical things that are simulated, and interactions are the events that occur in simulations, such as detonations and collisions.

The OMT is comprised of four tables:

- Class Structure Table: Records subclass-superclass relations between different types of federates/federation objects
- Object Interaction Table: Records the types of interactions possible between different classes of objects, their affected attributes and the interaction parameters.
- Attribute/Parameter Table: Specifies features of the public attributes of objects and the parameters of the interactions
- Lexicon: Defines all of terms used in tables

Runtime Infrastructure

As previously stated, the RTI's primary function is that of a data distribution mechanism. Federates send information through the RTI which distributes the information to the appropriate parties. The RTI does not maintain information about the state of the federation. Nor does it handle any semantics associated with the interaction between the federates, such as what coordinate system to use, what happens during a collision, or how to dead-reckon remote vehicles. Also, the RTI does not specify the exact byte layout of data sent across the network.

The RTI provides a common set of services to the federates. They can be divided into six categories:

- Federation Management: Handles the creation, dynamic control, modification, and deletion of a federation execution.
- Declaration Management: Enables federates to declare to the RTI their desire to generate (publish) and receive (subscribe/reflect) object state and interaction information. Federates can subscribe to only the objects they want (or have the capability) to receive.

- Object Management: Enables the creation, modification, and deletion of objects and interactions. These services comprise most of the network traffic during runtime.
- Ownership Management: Allows federates to transfer ownership of object attributes to other participants in the simulation.
- Time Management: Provides useful services for setting, synchronizing, and modifying simulation clocks. Time Management services are tightly coupled with the Object Management services so that state updates and interactions are distributed in a timely and ordered fashion.
- Data Distribution Management: Federates can provide conditions governing when to start or stop transmitting and receiving certain pieces of data.

TRANSITIONING FROM DIS TO HLA

As the training and simulation industry moves toward HLA compliance, current DIS simulators will need to be updated in order to keep from becoming obsolete. There are four techniques for making the DIS to HLA transition—translator, wrapper, native, and protocol interface unit (PIU). Some of these techniques are simpler and more cost-effective than others, and each has its own advantages and disadvantages. Each of the techniques are discussed below.

Translator

Using this technique, a separate application, often another computer, is placed on the network to translate network traffic between the different protocols.

A translator requires no software modification to the simulator, but because data must travel though this extra piece of hardware, the simulator's latency increases by roughly a factor of ten. Having all traffic pass through one computer is risky since it puts a single point of failure into an otherwise distributed system. The translator does permit limited forward and backward compatibility, but limits the scalability and flexibility of the simulator. Also, when using a translator, the simulator cannot take advantage of future HLA features.

Wrapper

With a wrapper approach, software is added underneath the simulation's DIS interface to translate the data from the old protocol (DIS) to the new protocol (HLA) just before it is sent and to translate the data from HLA to DIS just after it is received.

Unlike the translator, a wrapper does not require additional hardware. All the changes are made via limited modification to the simulator's software. However, forward and backward compatibility requires further software changes, and like the translator, the wrapper does not allow the simulator to take advantage of HLA specific features.

Native

Creating a native HLA simulation implies that all interfaces to the network are contained within the simulation software.

A native HLA simulator can take full advantage of all HLA features. However, these advantages come at the expense of huge software modifications at the initial transition and then additional modifications for any future protocol changes. Also, there is no backward compatibility.

Protocol Interface Unit

The simulation interfaces with the network via a software system known as a Protocol Interface Unit (PIU). A good PIU, such as MÄK Technologies' VR-Link™, will have one API (the simulation's interface) which supports all the features in both protocols, DIS and HLA. A less capable PIU will limit functionality to the lowest common denominator, and the simulator will be unable to take advantage of any features unique to either protocol.

The PIU approach may be the best technique to update a DIS simulator to HLA. It provides an easy upgrade path to HLA, while maintaining backward compatibility with DIS. Using a PIU also permits a simulator to switch among different FOM's within HLA and even different versions of the DIS protocol. A PIU requires only minimal modifications to the simulation software and provides the most flexibility when designing a new simulation. On the downside PIUs can be complex and expensive to write and maintain.

CONCLUSION

DIS has proven to be a valuable tool to train soldiers, perform weapons assessment, and conduct mission rehearsals. As DIS moves into the HLA paradigm, it provides an even more powerful and flexible framework for the modeling and simulation

community. Over the next four years, as DIS simulations transition to take advantage of the HLA, their value will increase through the integration of a wide range of existing and planned simulations.

We've demonstrated several techniques available to transition DIS-compatible simulations to this new paradigm. One of these, the Protocol Interface Unit, stands out as offering the most flexibility, while incurring the least amount of risk and cost. By using a PIU, such as VR-Link, simulations can maintain their DIS compliance vital to ongoing projects, while providing a clear upgrade path for future HLA simulation programs

REFERENCES

Department of Defense. 1996a. *High Level Architecture for Simulations Version 0.5 Interface Specification.*

Department of Defense. 1996b. *High Level Architecture Object Model Template Version 0.3.*

Institute of Electrical and Electronics Engineers. 1995. *IEEE Standard for Distributed Interactive Simulation - Application Protocols.* IEEE Publishing. NY.

AUTHOR BIOGRAPHIES

DEBORAH A. FULLFORD is a member of the technical staff at MäK Technologies. She has a B.S and M.S in Aeronautical Engineering. She is a member of the Society of Computer Simulation and has served as both Session and Panel Chair for the Distributed Simulation Track of the Summer Computer Simulation Conference and the Multi-Simulation Conference.

ALGORITHM OPTIMIZATION FOR REAL-TIME SIMULATION

By
Harry Heaton and Jennifer DeVilbiss
Science Applications International Corporation
Beavercreek, Ohio 45431

ABSTRACT

The advent of high performance microprocessor-based processing architectures encourages ever-greater fidelity and response requirements for man-in-the-loop simulation. In spite of processor clock speeds in the hundreds of megahertz, achieving this level of performance in an affordable system requires the application of highly optimized algorithms.

We review specific real-time optimization examples used in the development the F-111 Digital Radar LandMass Simulation (DRLMS), a real-time ground mapping radar simulator written in entirely in Ada 95, and targeted for high-end, general purpose RISC processors. Optimization techniques familiar to circa 1980 avionics programmers, updated to take advantage, or avoid the pitfalls, of modern RISC architectures, are examined. Topics include polynomial approximations of complex functions, the effects of instruction pipelining, and organizing high level code to take advantage of hardware-level cache optimization.

F-111C DRLMS OVERVIEW

The F-111C DRLMS consists of an Attack Radar System and 2 Terrain Following Radars. Within each radar system, is a SAIC proprietary core radar simulation that is called RADSIM* . RADSIM is a high fidelity radar simulation implemented exclusively in software that provides the fidelity and flexibility to a Digital Radar Landmass Simulation (DRLMS) system. RADSIM includes the fundamental algorithms and processes necessary to produce radar imagery. RADSIM will use a digital database created from overhead imagery; Digital Terrain Elevation Data (DTED) and Digital Feature Analysis Data (DFAD), and other sources using PhotoMap database generation toolset.

The DRLMS runs on an 8 x 200 MHz processor Silicon Graphics Challenge. RADSIM is split between 2 processors for the Attack Radar System and runs on 1 processor for each of the Terrain Following Radars. The DRLMS executive, controllers, views, and other models run on the remaining processors.

The F111-C DRLMS and RADSIM were designed using the Booch object-oriented methodology and the standard interfaces between the objects were loosely based upon the Software Structural Model. Figure 1 represents the software architecture of the SAIC DRLMS. The SAIC-patented RADSIM is contained with the Models Class.

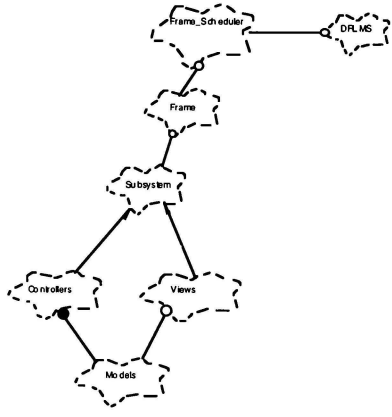


Figure 1: DRLMS Software Architecture

Copyright ©1997 by Jennifer DeVilbiss and Harry Heaton. Published by the American Institute of Aeronautics and Astronautics, Inc. with permission.

The Controllers class allows interfaces to be built that are customized to a particular simulator's control interfaces. The Views Class output displays that match the modeled radar systems. The Models Class contains the Core radar simulation code (i.e., RADSIM), as well as, any additional models that were required (e.g., the antenna model). The Executive level classes initialize the DRLMS components, start the real-time processing of the DRLMS, and handle the timing of the real-time activities.

SOFTWARE PERFORMANCE ENHANCEMENTS

The F111-C was designed and coded using modern object-oriented techniques, such as inheritance, dynamic binding, and information hiding. For instance the combination of inheritance and dynamic binding make it possible to add entirely new capabilties to the simulation without changing a single line of the executive. The software was also however, coded for performance. Bottlenecks and tight loops were identified and, where appropriate, optimized to maintain the maximum performace capability.

By design, RISC processors obtain their maximum performance executing an uninterrupted stream of simple, pipelined instructions produced by an optimizing compiler. The techniques discussed below are approaches to source code optimization that increase the degree to which this ideal is realized. Implementing the techniques requires that the software engineer understand the organization and limitations of both the target processor and the compiler, tailoring the source to best match the features of both.

RISC Instructions

There were some basic performance enhancements done that were based off the number of instructions needed to execute the lines of code. For example, multiplication was used instead of division wherever possible. On the MIPS† R4000‡ instruction set machines a divide takes 23 instructions for single precision, 36 for double verses a multiply which takes 7 instructions for single precision and 8 for double.

Square root and other math functions were also avoided due to their increased instruction processing time. Certain math functions such as square root and divide also tie up critical registers which can stall the pipeline. The number of "if" statements were also minimized in tight processing loops. Since the RISC architecture prefetches instructions, a branch on a conditional statement (if) does not take advantage of this prefetching. If the wrong branch is prefetched, then the pipeline backs up while it fetches the correct instructions.

Caching

Large arrays were needed for the processing of the image and terrain data (up to 72000 items). These arrays were slices of data, or spokes, needed to generate the radar displays. Logically, these arrays were as follows but physically they were one-dimensional arrays.

```
-------------
-333333-
-222222-
-111111-
-------------
```

These arrays sizes were kept small to the minimize cache refreshes. Smaller intermediate arrays were used wherever possible to also reduce cache refreshes while accessing the data. Individual rows of data are processed one at a time with an array the size of the row instead of the total array size (row times the number of columns).

Multi -Processing

Due to the fact that the Attack Radar System needed to process large array sizes to fill the radar display, the RADSIM code was split between two processors. This ensured that it would keep within the allocated time frame. While data was being retrieved from the portion of the database that was kept in memory (the patch) on one processor, any preprocessing of data that could be accomplished before the radar effects were generated were executed on the other processor. Once these two preprocessing operations were complete, the arrays were split in half horizontally with each half processed on each processor. Due to the fact that the algorithm for the terrain processing accessed the data in columns, the array for the terrain data was split in half vertically as opposed to horizontally. This kept the processor from

constantly swapping cache by reducing the frequency that the index fell outside of the cache line length.

Array Contents

Another consideration is the data type defined for the large arrays. Whenever possible, integer arrays were used to keep addressing of the data on even boundaries for quicker access. Although, this had to be weighted against the fact that byte arrays take up less memory and, therefore, more data could be kept in cache. Also, single and double precision floating point arrays were minimized due to the increased processing time associated with them.

Array Addressing

For the F-111C DRLMS, 2-3 milliseconds were shaved off the update time of the core by using an array addressing scheme that is shown in the following example. This reduced time was critical in keeping the DRLMS update rate below the 32 MHz frame time.

Ada95 Example:

```
type int_ptr is access integer;

function Address_to_Access is new

  Unchecked_Conversion(system.address,
                       int_ptr);
 function Access_to_Address is new
   Unchecked_Conversion(int_ptr,
system.address);

 function Integer_to_Address is new
    Unchecked_Conversion(Integer,
                         system.address);

  function Address_to_Integer is new
     Unchecked_Conversion(system.address,
                          Integer);

Pixel_Storage : constant := 4;
Bitmap        : array (1 .. 20000) of integer;
Bitmap_Ptr : system.address :=
             Bitmap(1)'address;
procedure test is
begin
Address_To_Access(Spoke_Bitmap_Ptr).all := 5;
Spoke_Bitmap_Ptr :=
    Integer_To_Address(
      Address_To_Integer(Spoke_Bitmap_Ptr) +
        Pixel_Storage);
end test;
```

This array addressing scheme increases the performance in tight loops by allowing the calculation of the next array position to be done by adding the appropriate storage size instead the base address being recalculated for each array position. It should be noted, however, that this type of coding skirts the compile and run-time checking by going down to the lowest level to address the array. For the F111-C DRLMS project, the development of the code was done using normal array addressing and testing was done to ensure that the code was working correctly. Software performance enhancements such as this example was then added and retested.

ALGORITHM OPTIMIZATION TECHNIQUES

For many purposes, the speed of modern micro-processors makes the use of models based directly on first-principles a reality. While simplifying the validation of algorithms, these approaches generally do not produce the most efficient real-time code. The techniques below are familiar to a generation of programmers who hand-assembled code for real-time systems. In spite of RISC processors with speeds above 200 Mhz, these techniques were used to refine critical real-time algorithms for the F-111C DRLMS.

Polynomial approximations

Non-linear functions can often be approximated over a limited range by a polynomial, at a substantial savings in computational demand. Consider the example of computing the curvature of the earth, a necessary computation for many sensor simulations. For a spherical earth, this may be computed along any line of sight from the equation of a circle. In Cartesian coordinates, the computation of the loss in height may be derived

from the equation of a circle, with center at the origin:

$$r = \sqrt{\left(x^2 + y^2\right)} \qquad \text{(eq. 1)}$$

re-arranging and solving for the square root and subtracting from the radius finds our loss in height (h):

$$h = r - \sqrt{\left(r^2 - x^2\right)} \qquad \text{(eq. 2)}$$

where:

r = Earth's radius
h = loss in height (y axis)
x = horizontal range from origin

If computed for a single point, this computation is trivial. If computed for every point in a simulated sensor image however, the presence of a square root greatly increases the computational burden.

An alternative for this computation is a polynomial curve fit, which takes the form:

$$h = m_0 + m_1 * x + m_2 * x^2 \ldots + m_n * x^n \qquad \text{(eq. 3)}$$

where:

m = polynomial coefficient
n = order of the polynomial
h = loss in height (y axis)
x = horizontal range from origin

Generating the coefficients and determining the order of the curve fit may be accomplished by several means. This author finds it most convenient to generate the data for the exact solution, then fit the curve using a commercial scientific graphing package, such as KaleidaGraph [§]. In using such graphing packages, simple controls allow changes to the type and order, until a suitable fit is obtained.

This approach allows a simple, interactive method to view the result of the curve fit and examine the minimum order of the polynomial necessary to obtain an adequate fit. Figure 2 and 3 depict the plotted values from the rigorous computation, and the line for the third-order curve fit. Table 1 lists the polynomial coefficients.

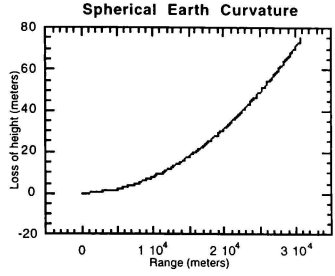


Figure 2: Rigorous Solution (Equation 1)

Table 1: Equation 3 Coefficients

Curve fit Term	Value
M0	-6.1592e-06
M1	4.03e-09
M2	7.8495e-08
M3	2.982e-17
R (Correlation)	1.0

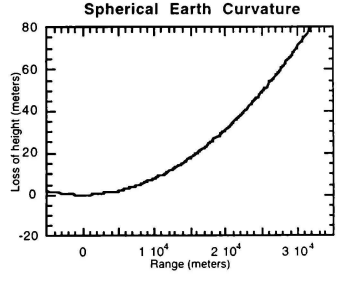


Figure 3: Polynomial Fit (Equation 3)

Clearly, for this example, the results generated by the curve fit are equivalent to the rigorous solution. But does it help our real-time performance?

If we examine the expected instructions and their cycle times between the rigorous implementation and the polynomial implementation we find the cycle times depicted in Table 2 and 3, based on published times for double-precision instructions for the MIPS R4000 FPU.

Table 2: Apparent FPU Instruction Cycles for Rigorous Solution

Instruction	Qty	Max. Cycles	Total Cycles
Subtract	2	4	8
Multiply	1	8	8
Square root	1	112	112
Total			**128**

Table 3: Apparent FPU Instruction Cycles for Polynomial Curve Fit

Instruction	Qty	Max. Cycles	Total Cycles
Addition	3	4	12
Multiply	6	8	48
Total			**60**

Based on a simple counting of clock cycles, it appears that the third-order curve fit will execute in about one-half of the time of the rigorous solution. However, this summation of clock cycles does not take into account the instruction pipelining and resource requirements in the FPU. Actual benchmarks, obtained on a MIPS- R4400[1] operating at 250 MHz, show a different story:

Table 4: Performance Benchmarks

Method	Timing
Rigorous	5.9959E-07
Polynomial	3.9694E-09

The polynomial approximation benchmarked over 150 times faster than the rigorous solution. Since each stage of the polynomial adds only a two multiplies and an addition, even high-order polynomials, describing complex functions can be computed, still with substantial savings. For an inner-loop operation that is repeated nearly half a million times a second, these are critical savings.

Two reasons for this disparate performance are possible: One, the sequence of simple operations in the polynomial fit allows the optimizing compiler to take advantage of instruction pipelining, or two, the compiler does not implement the hardware square root instruction. Some compilers make calls to software math libraries rather than to the far faster FPU instructions.

Polynomial curve fitting is also applicable to modeling data available only in graphic form, such as empirical performance data, especially when such data is non-linear. This can be superior to a simple table look-up of such data, especially if interpolation is required to extract intermediate points. The polynomial curve fit avoids the pipeline restarts required to service branch instructions. Figure 4 depict the empirical data points for radome refraction versus antenna elevation angle, along with the polynomial curve fit to the data. Thus even for strongly non-linear data, this method can provide good results, even if more than one polynomial is required to depict the data .

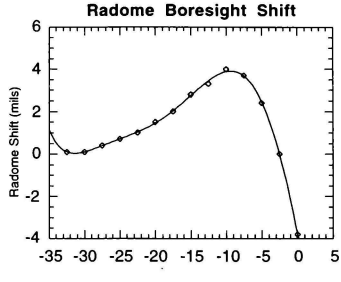


Figure 4: Polynomial fit to empirical data set.

Changing Paradigms

Profiling the performance of the code is an essential step in determining which routines will benefit most from algorithm changes. On a previous radar simulator project, a single inner-loop terrain occulting routine was occupying 16 ms of a 40 ms total budget. Profiling the routine revealed that 1/3 of the routine's time was spent in the

arctangent function call. This call was essential in this design, in which all comparisons were made in angular units.

The original algorithm determined both the grazing angles of rays from the source to subsequent terrain posts in a gridded representation of terrain, as well as the post-to-post slope of the terrain. Comparing the incident angle of the rays from the last non-occulted post to the next current post (working away from the illumination source) determined whether the current post was occulted. Further, comparison of the slope of the incident ray to the slope of the terrain at the intersection point determined the degree of perpendicularly of the terrain to the ray.

Reducing the execution time of this routine was essential to meeting execution time requirements. Initial attempts at performance enhancement attempted to preserve the underlying algorithm, concentrating instead on data type and structural changes. These changes provided modest performance gains, but not the magnitude required.

The first significant algorithm modification changed the comparison of the incident rays to slope, rather than angle, for the determination of occulting. This change eliminated the arc tangent, but required changes throughout the routine to accommodate the change from angle to slope. Still, the performance gain was substantial, as depicted in Table 5.

Curiously, a change from floating-point operations to scaled integer operations actually resulted in an increase in execution time. In this case, the modest difference between floating point and integer instruction timing in the RISC processor was offset by the added complexity of scaled integer operations. It is also possible that moving operations from the FPU pipeline into the integer pipeline reduced inherent parallelism.

Table 5 Terrain Timing

Change to Algorithm	Time (s)	ΔTime (s)
1) Baseline	0.0162	0.00
2) Slope instead of angle (eliminates arctan call)	0.0120	0.042
3) Slope, (same as 2), but with scope reduction.	0.0119	0.0001

A second simplification eliminated the computation of the terrain slope used to determine perpendicularly. As this algorithm operated on a gridded database, it was noted that the rate of change of the slope of the incident rays between adjacent posts followed a fixed function if the terrain was flat:

$$m_n = m_{n-1} \div \left(1 + \frac{1}{1n}\right) \qquad \text{(eq. 4)}$$

where:

m = depression slope of the incident ray

n = number of terrain posts from post zero

Thus the slope of an incident ray can be predicted, based on the previous ray. If the slope of the ray is less than or greater than the prediction, then the underlying terrain is rising or falling respectively. Extending this check allows the perpendicularity of the terrain to be checked without actually computing the slope of the underlying terrain.

The resulting algorithm eliminated several branch instructions required by the previous approach. This resulted in an additional 4 milliseconds reduction in processing time over the baseline, reducing by more than half the original processing time for the occulting routine. Overall, the combination of algorithm and structural optimizations employed on the F-111C DRLMS results in nearly seven times the data throughput in comparison to earlier RADSIM™ based radar simulators.

CONCLUSIONS

Increasing processor performance is offset by increasing customer expectations for real-time simulations. Real-time design requirements challenge the software engineer to understand the underlying operations of the target hardware in order to structure high-level code that can be effectively optimized by modern compilers.

The authors acknowledge Mr. Tom Haberlandt and Mr. James Hopper for their contributions in both the design of the RADSIM software and their helpful suggestions for this paper.

REFERENCES

1) Heinrich, Joe. *MIPS R4000 User's Manual*, Englewood Cliffs, NJ: PTR Prentice Hall, 1993.

---

* RADSIM is a trademark of Science Applications International Corporation.

† MIPS is a registered trademark of MIPS Technologies, Inc.

‡ R4000 is a registered trademark of MIPS Technologies, Inc.

§ KaleidaGraph is a trademark of Abelbeck Software.

¶ R4400 is a registered trademark of MIPS Technologies, Inc.

AIAA-97-3597

COMPARISON OF SCATTERING MODELS FOR PREDICTING RADAR MULTIPATH EFFECTS OVER THE SEA

Domenic Bucco,* Defence Science and Technology Organisation (DSTO), Salisbury, Australia.
James M. Chisholm,† Naval Surface Warfare Center, Dahlgren Division (NSWCDD), Dahlgren, Virginia, USA.

Abstract

Two mathematical models for predicting the effects of multipath scattering over the sea surface are presented and compared. The first model is a phenomenological model based on empirical results and observations of the scattering phenomena reported by Beard, et al, during the 1950's and 1960's. The other model is a theoretical model based on physical principles and derived using the Kirchoff assumptions. The key features of both models are described and the limitations of each are identified.

1. Introduction

An important and often significant modeling consideration when assessing the performance of monopulse tracking radars in an oceanic environment is multipath. Applications of naval interest include the performance of RF guided missiles against sea skimming targets and the multipath effects on the surveillance and tracking of low altitude targets over the sea by shipborne radars. Models of the multipath scattering phenomena that have appeared in the literature tend to follow a common theme in which the total reflected signal is considered to be the addition of coherent (specular) and incoherent (diffuse) terms. However, within this theme, two different approaches have apparently emerged. One of these approaches to multipath modeling is based purely on the phenomenological and empirical relations observed from over-sea experimental data[1-4] and amounts to the calculation of the total field strength at the receiver via vector addition of the direct and reflected fields. In this approach, the point of origin of the diffuse scattering vector is taken as the specular reflection point. The other popular approach to multipath modeling follows from published theoretical work[5-7] and is based on incoherent power addition instead of vector addition. In this case, mean power reflection coefficients are determined to characterize the specular and diffuse components. The total diffuse power, in turn, is computed by considering the division of the glistening surface into a sequence of trapezoidal facets, with each facet acting like a small mirror in reflecting the incident wave according to the laws of geometrical optics. In this diffuse scattering model, the sea surface is completely defined by the distribution of the slopes of these elementary facets.

In this paper, we compare two RF scattering models that are being utilized for the prediction and simulation of multipath effects over the sea. One of these models (model-A) is based on the phenomenological vector approach proposed by Beard, Katz and Spetner[1-3] following their observations of over-sea experimental measurements. This vector model was subsequently enhanced by Northam[4] into a stochastic simulation of the multipath effects using Gauss-Markov processes. Model-A has been implemented in both FORTRAN and MATRIXx and is currently being used in the Weapons Systems Division at DSTO for predicting over-sea multipath effects in support of missile performance studies.

The other model (model-B) is based on the theory of scattering of electromagnetic waves from a rough surface as presented by Beckmann and Spizzichino,[5] Barrick,[6] and later modified by Barton.[7] The multipath simulation model has been implemented in FORTRAN and is being utilized by the Weapons Systems Department at NSWCDD in performance studies of semi-active missiles.

The purpose of this paper is to report on the preliminary results of a comparative study of the two

---

* Senior Research Scientist, Weapons Systems Division, Member AIAA.
† Aerospace Engineer, Weapons Systems Department.
This paper is declared a work of the U. S. Government and is not subject to copyright protection in the United States.

multipath models. First, a functional description of each model is outlined. This is then followed by a discussion on the qualitative comparison of various features and limitations of each model.

2. Model-A

Model-A is based on the concept of vectors representing the electric field strength at the missile receiver. This field strength is modeled as the sum of two components, a direct path vector and a reflected path vector. Consider the vector model of reflection shown in Figure 1.

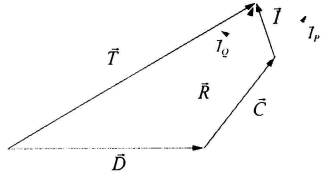


Figure 1. Receive Signal Vector Model

Here, the total signal ***T*** is given by the vector sum

$$\vec{T} = \vec{D} + \vec{R} \qquad (1)$$

where the direct signal ***D*** (the signal received in the absence of multipath effects) is taken as the reference vector and is assumed constant in both amplitude and phase. According to Beard, Katz, and Spetner[1-3] the reflected signal is composed of a coherent (specular) component, ***C***, whose amplitude and phase are fixed by a given geometry and sea state, and an incoherent (diffuse) component, ***I***, having random amplitude and phase. Thus, the total signal received by the missile receiver is

$$\vec{T} = \vec{D} + \vec{C} + \vec{I} \qquad (2)$$

The coherent component is the specularly reflected vector which would be received if the sea were perfectly smooth, modified by a rough sea scattering coefficient. The incoherent component accounts for the time varying effects of the sea on the receive signal, in particular, on the signal spectrum.

In the following, we shall assume an X-band radar utilizing linear vertical polarization with no change in polarization due to reflection, no variation in sea state during a simulation, homogeneous atmospheric conditions, and a flat earth geometry. Consider the multipath geometry depicted in Figure 2, in which the transmitter and receiver are fixed relative to the earth, a distance R apart, and at heights above the mean sea level of H_T and H_R, respectively.

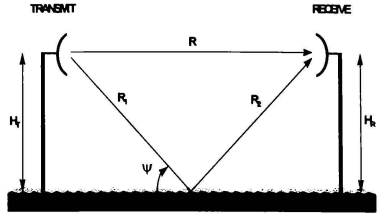


Figure 2. Basic Multipath Geometry

If we assume that the geometry does not change with time (moving platform multipath will be considered later), then clearly the only significant variation in the received signal is due to sea surface motion. Consequently, the sea surface motion can be viewed as directly controlling the variation of the composite signal (direct plus reflected) received at the antenna. Following Northam[4], we use a characterisation of the second order properties of the sea surface motion to determine the received signal's second order properties. Thus, for our multipath simulation, the sea surface motion can be adequately characterised by a simple representation of the prevailing wave height spectrum and a sea surface roughness parameter g. These two observables are considered to be the most fundamental parameters of the model. Here, the roughness parameter g is defined by

$$g = \frac{\sigma_h \sin(\psi)}{\lambda} \qquad (3)$$

where σ_h is the RMS wave height,
ψ is the grazing angle,
and λ is the radar wavelength,
while the Neumann spectrum[4] is used in the simulation as a model of the prevailing wave height spectrum.

The following sections describe in detail the various components of the phenomenological vector model outlined above and how they have been implemented in the present multipath simulation model.

2.1 Coherent Component

The equation describing the coherent component is:

$$\vec{C} = |\vec{D}|\Gamma\rho_c \cdot e^{-j2\pi\Delta R/\lambda} \quad (4)$$

where $\vec{D}$ is the direct signal,
Γ is the smooth sea (Fresnel) reflection coefficient (complex),
ρ_c is the coherent scattering coefficient,
ΔR is the path-length difference between the direct and reflected paths.

For linear vertical polarisation of the transmitted wave, Γ has the form[5]

$$\Gamma = \frac{\varepsilon_c \sin\psi - \sqrt{\varepsilon_c - \cos^2\psi}}{\varepsilon_c \sin\psi + \sqrt{\varepsilon_c - \cos^2\psi}} \quad (5)$$

where ε_c is the complex dielectric constant defined by the equation

$$\varepsilon_c = \varepsilon - j60\lambda\sigma \quad (6)$$

where ε denotes the sea's dielectric constant, and σ is the conductivity of the sea in mho/m. (Typical values of these parameters for sea water are: $\sigma = 4.3$ mho/m and $\varepsilon = 80$.)

The coherent scattering coefficient ρ_c is based on Beard's experimental data[3,4] and is given by

$$\rho_c = \begin{cases} e^{-2(2\pi g)^2}, & 0.0 \le g \le 0.1 \\ \dfrac{0.812537}{1+2(2\pi g)^2}, & 0.1 < g \le 0.3 \end{cases} \quad (7)$$

Note that this expression for ρ_c has been validated for the cases where $g \le 0.3$, as shown in Figure 3. Because there are no known validated models of ρ_c available for instances where $g > 0.3$, we have followed Northam's approach in the present simulation by extending the upper limit to include all values of g.

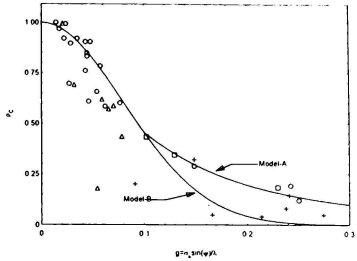


Figure 3. Coherent field versus apparent ocean roughness

2.2 Incoherent Component

The incoherent component is stochastic in nature and is assumed to be the resultant sum of vectors from many, independent, random scatterers. Therefore, $\boldsymbol{I}$ is represented as the sum of two zero mean Gaussian processes, $\boldsymbol{I_P}$ and $\boldsymbol{I_Q}$, which are orthogonal, as depicted in Figure 1. For convenience, the phase of $\boldsymbol{I_P}$ is assumed to be that of $\boldsymbol{C}$. In addition, we assume that $\boldsymbol{I_P}$ and $\boldsymbol{I_Q}$ are independent and that their power spectral densities are identical.[4] Consequently, the incoherent component can be expressed by

$$\vec{I} = |\vec{D}|\Gamma(\vec{I_P} + \vec{I_Q}) \quad (8)$$

The standard deviation of $\boldsymbol{I_P}$ and $\boldsymbol{I_Q}$ are identical and are determined by the incoherent scattering coefficient ρ_I. Therefore, for simulation purposes, the expression

$$\vec{I} = |\vec{D}||\Gamma|\rho_I(I_1 + jI_2) \quad (9)$$

is used to generate $\boldsymbol{I}$. In the above expression, $\boldsymbol{I_1}$ and $\boldsymbol{I_2}$ are independent Gaussian processes having zero mean and unity variance, that is, $I_1, I_2 \sim N(0,1)$.

In an attempt to curve fit the experimental data of Beard, et al[1-3] for multipath simulation purposes, Northam[4] proposed a linear model of ρ_I :

$$\rho_I = \begin{cases} 3.75g, & 0 \le g \le 0.1 \\ 0.4625 - 0.875g, & 0.1 < g \le 0.5 \\ 0.025, & g > 0.5 \end{cases} \quad (10)$$

Figure 4 illustrates the relationship of the three-part linear model of Equation (10) with Beard's experimental data.

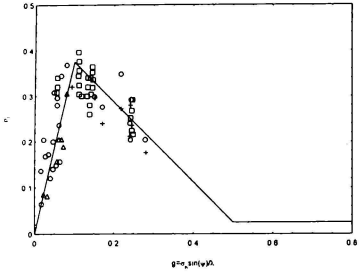


Figure 4. Incoherent scattering coefficient.

Note the good agreement between the linear model and the experimental data for the region $g \le 0.3$.

2.3 Received Signal Spectrum

Up to this point, we have described only the first order properties of the total received signal. To complete the second order, stochastic model, we need a description of the received signal spectrum or, equivalently, the autocorrelation function of the received signal (RS).

It is clear from Figure 2 that, for a fixed transmitter/receiver geometry, the fluctuations in the RS are due to the motion of the sea surface. Therefore, it would follow intuitively that the various characteristics of such RS fluctuations should be directly related to the associated characteristics of the sea surface variations. In this regard, Beard and Katz[2] have presented valuable data which quantitatively verifies this relationship. Moreover, their experimental findings provide pertinent information for deducing the approximate shape of the RS spectrum given only a knowledge of the roughness parameter and the peak frequency in the ocean wave height (WH) spectrum. As these experiments were carried out at X-band and for vertical polarization of the transmitted signal, the results are applicable here. Basically, the approximate results found experimentally[2,4] are:

(a) the upper 3 dB frequency of the RS spectrum is linearly related to the peak frequency of the WH spectrum for values of the roughness parameter greater than 0.1, and

(b) the RS spectrum centre frequency is approximately equal to the WH spectrum centre frequency,

Furthermore, by using a series expansion of the magnitude of the total receive signal $\boldsymbol{T}$ (Figure 1), Beard and Katz[2] have shown that

(c) when $\boldsymbol{D}$ and $\boldsymbol{C}$ are in phase, to first order, the spectrum of $|\boldsymbol{I_P}|$ is equal to the spectrum of $|\boldsymbol{T}|$.

Thus, the first step in this approach is to choose a representative model for the WH spectrum. Although various models for such spectra exist, for our purposes we need only a model that will yield an estimate of the WH peak frequency, f_W, for a given sea state or, equivalently, a given wind speed. A useful analytic form for typical WH spectra has been proposed by Neumann.[8] Beard and Katz[2] have shown that the Neumann spectrum compares favourably with experimental results.

The Neumann spectrum is described by

$$A^2(\omega) = \frac{\pi}{2} C \omega^{-6} e^{(-2g_e^2 \omega^{-2} V_W^{-2})} \quad (11)$$

where $A^2(\omega)$ is the power spectrum,
ω is the frequency (rad/s),
g_e is the acceleration due to gravity,
V_W is the wind speed (ms^{-1}),
and C is a constant, $C = 3.05\ m^2\ s^{-5}$.

From Equation (11), the WH center frequency can be derived

$$f_W = \frac{g_e}{\sqrt{6}\,\pi V_W} \quad (12)$$

Next, in order to calculate the roughness parameter $\boldsymbol{g}$ for a given sea state, we need an estimate of the RMS wave height σ_h. This is determined by computing the total energy $\boldsymbol{E}$ in a fully aroused sea. The sea is considered fully aroused here when, for a given sea

state and hence wind speed, all frequencies contribute to the total energy.

Based on the Neumann spectrum, the total energy in a fully aroused sea is given by

$$E = 3\left(\frac{\pi}{2}\right)^{\frac{1}{2}} C^{'}\left(\frac{V_W}{2g_e}\right)^5 \qquad (13)$$

from which the RMS wave height is determined as

$$\sigma_h = 1.765 \times 10^{-3}\, V_W^{5/2} \qquad (14)$$

Thus, given the prevailing sea state, and hence wind speed, Equations (12) and (14) above are used in the simulation to compute the roughness parameter and the peak frequency of the WH spectrum.

The corresponding RS spectrum characteristics can now be determined. From result (a) above, and by adopting Northam's extrapolation approach for low values of g, the ratio of the frequency (Hz) of the RS spectrum at which the relative power has been reduced to one half (the 3-dB frequency point) and the peak frequency (Hz) of the WH spectrum is given by

$$\frac{f_{RS}(0.5)}{f_W} = \begin{cases} C_0 + C_1 g^{C_2}, & 0 \le g < 0.1 \\ C_0 + C_1 g, & 0.1 \le g \end{cases} \qquad (15)$$

where C_0, C_1 and C_2 are the constant coefficients given in Table 1. Equation (15) has been plotted in Figure 5 together with experimental data.[2]

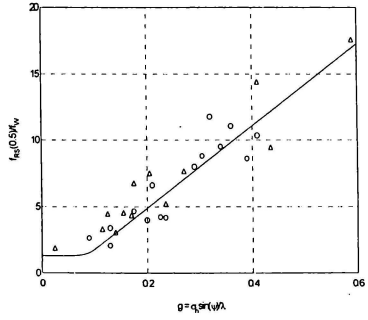


Figure 5. Calculation of $f_{RS}(0.5)$ from sea roughness and wave spectrum centre frequency.

Coefficient	$0 \le g < 0.1$	$0.1 \le g$
C_0	1.30	-1.295
C_1	7.744×10^5	30.95
C_2	6.19	-

Table 1. Value of coefficients associated with Equation (15)

Additionally, from (b) above, we have

$$f_c \approx f_W \qquad (16)$$

where f_c denotes the centre frequency (Hz) of the RS spectrum. Consequently, the bandwidth, BW, of the RS spectrum can be calculated.

Finally, in order to calculate the above RS parameters, we need to specify the sea state at the beginning of a simulation run. The surface wind speed, V_W, is then calculated in accordance with Table 2. Additionally, Table 2 also provides the RMS wave height, σ_h, and the peak frequency of the Neumann wave height spectrum, f_W, as a function of sea state.

Sea State	Wind Speed (m/s)	RMS Wave Height (m)	f_W (Hz)
0	0	0	-
1	2.5	0.018	0.509
2	4.5	0.08	0.283
3	6.8	0.21	0.187
4	9.4	0.48	0.135
5	12.3	0.95	0.104

Table 2. Variation of sea surface parameters with sea state.

2.4 Generation of the RS Spectrum

Our approach here is to generate the I_P and I_Q processes having bandpass spectra each with centre frequency f_c (from Equation (16) above) and bandwidth BW (from Equation (15) above), and variance ρ_I (from section 2.2). These processes will be wide-sense stationary, zero-mean Gaussian and independent. Moreover, because of the bandpass

nature of the processes, they will have autocorrelation functions of the form

$$R(\tau) = e^{-\alpha\tau}\cos(\omega\tau) \tag{17}$$

where α and ω have units of rad/s and are determined from f_c and BW.

Let us consider first the exponential part of the autocorrelation function. A digital method of generating a first-order, Gauss-Markov sequence X_k with autocorrelation function $R_X(\tau) = e^{-\alpha\tau}$, using a step size Δt, is given by[4]

$$X_k = AX_{k-1} + BW_k \tag{18}$$

where, for each k, the input W_k is a Gaussian, white noise sequence having zero mean and unity variance, that is, $W_k \sim N(0,1)$, and the initial conditions for X_k are $E\{X_0\} = 0$ and $E\{X_0^2\} = 1$. The coefficients A and B are given by

$$A = e^{-\alpha\Delta t} \tag{19}$$

$$B = \sqrt{1 - A^2} \tag{20}$$

To complete the RS process generation, we need to transform X_k, as computed above, to a process with a bandpass spectrum. This is achieved by defining another sequence Y_k such that

$$Y_k = X_k^1\cos(\omega k\Delta t) + X_k^2\sin(\omega k\Delta t) \tag{21}$$

where now X_k^1 and X_k^2 are two independent Gauss-Markov sequences generated according to equation (18) above, and hence are stationary and will have the same autocorrelation function $R_X(\tau)$. The Y_k process is also Gaussian because it is a memoryless linear transformation of Gaussian processes. Furthermore, it is stationary with the desired autocorrelation function as given by Equation (17).

Thus, in the digital simulation, the generation procedure for the two incoherent processes $\boldsymbol{I_P}$ and $\boldsymbol{I_Q}$ is as follows:

$$I_{P,k} = \rho_I\left[X_k^1\cos(\omega\, k\,\Delta t) + X_k^2\sin(\omega\, k\,\Delta t)\right] \tag{22}$$

$$I_{Q,k} = \rho_I\left[X_k^3\cos(\omega\, k\,\Delta t) + X_k^4\sin(\omega\, k\,\Delta t)\right] \tag{23}$$

2.5 Antenna Pattern Effects

If the transmit and receive antennas have direct path amplitude gains $G_{T,D}$ and $G_{R,D}$ respectively, and reflected path amplitude gains $G_{T,R}$ and $G_{R,R}$ respectively, then the total received electric field, including antenna gain effects, is represented by[4]

$$\vec{T} = \vec{D}\left[G_{R,D}G_{T,D} + G_{R,R}G_{T,R}F_{mp}\right] \tag{24}$$

where the complex multipath factor F_{mp} normalised to the direct signal $\vec{D}$ is defined as

$$F_{mp} = F_R + jF_I \tag{25}$$

It has been shown that this method is not valid when the antennas do not fully illuminate the first Fresnel zone on the sea surface.[3,4]

2.6 Moving Platform Multipath

A validated model for the incoherent spectrum that arises when the transmit and receive platforms are moving relative to one another has not yet been obtained.[4] However, Northam has proposed a heuristic model for this spectrum and the present simulation is based on this model.

To simulate the incoherent process for the moving platform scenario, two independent Gaussian processes having similar spectra are generated. These spectra are characterized by[4]

(1) a bandpass spectral shape,
(2) center frequency proportional to the ratio of the specular point velocity to a "dominant" sea wave length, and
(3) bandwidths determined as in the stationary platform model.

Characteristics (1) and (3) are generated using the procedure described previously. To estimate the suggested center frequency, we calculate the specular point velocity V_{sp} from geometry considerations and approximate the "dominant" sea wave length by the average "wave length" $\bar{\bar{L}}$ according to Kinsman,[8] namely,

$$\bar{\bar{L}} = \frac{g_e}{3\pi}\bar{\bar{T}}^2 \tag{26}$$

where the average "period" $\bar{\bar{T}}$ is defined by

$$\bar{\bar{T}} = 0.285\, V_W^{-1} \tag{27}$$

and V_W here denotes wind speed in knots. Thus, the center frequency of the incoherent processes is determined by

$$f_c = \frac{V_{sp}}{\bar{\bar{L}}} \tag{28}$$

According to Northam,[4] the specular point velocity V_{sp} is approximated by one half of the transmitter-receiver closing velocity owing to the assumption of small specular angles used in his applications. However, in the present simulation, we have derived a more general expression for the specular point velocity V_{sp} based on geometrical considerations. This expression is given by

$$V_{sp} = \sqrt{\dot{X}_{sp} + \dot{Y}_{sp}} \tag{29}$$

where

$$\dot{X}_{sp} = U_R - \frac{W_R(X_T - X_R)}{H_R + H_T} + \frac{H_R(U_T - U_R)}{H_R + H_T} + \frac{H_R(X_T - X_R)(W_T + W_R)}{(H_R + H_T)^2} \tag{30}$$

$$\dot{Y}_{sp} = V_R - \frac{W_R(Y_T - Y_R)}{H_R + H_T} + \frac{H_R(V_T - V_R)}{H_R + H_T} + \frac{H_R(Y_T - Y_R)(W_T + W_R)}{(H_R + H_T)^2} \tag{31}$$

In the equations above, the coordinates (X_T, Y_T, H_T) and (U_T, V_T, H_T) denote the transmitter position and velocity, respectively, while the coordinates (X_R, Y_R, H_R) and (U_R, Y_R, H_R) denote the position and velocity of the receiver, respectively.

3. Model-B

As discussed in the introduction, model-B is based on the theoretical work of Beckmann and Spizzichino,[5] which was later modified by Barton,[6] regarding the scattering of electromagnetic waves from a rough surface. The underlying assumption of this approach is that the field scattered by the rough sea surface is made up of two components: the specular reflection component and the diffuse scattering component. Model-B treats the specular and diffuse scattering from the sea surface as coherent and incoherent power sources. Thus, the total signal power received can be written as:

$$P_{Total} = P_D + P_{Spec} + P_{Diff} \tag{32}$$

where P_{Total} is the total signal power received,
P_D is the direct signal power,
P_{Spec} is the mean specularly reflected power,
and P_{Diff} is the mean diffuse scattered power.

The mean power contributions from the specular and the diffuse components are computed by using the mean specular and diffuse reflection coefficients associated with each type of scattering. The following sections describe the specular and diffuse reflection components in more detail.

3.1 Specular Reflection Component

Although specular reflection actually occurs over a small region of the sea surface called the first Fresnel zone (or more correctly, the first Fresnel ellipse), the specular reflection component is usually treated as a reflection from a single point on a perfectly smooth surface. This single point is referred to as the specular point. Geometric rays, as seen in Figure 2, are used to represent the incident and reflected RF signals. These rays obey the laws of geometrical optics, meaning that the angle of reflection is equal to the angle of incidence. Therefore, the specular reflection component is directional and its phase is coherent. This results in a reflection which has only one path to the receiving platform.

However, because the reflecting surface (in this case, the sea) is not perfectly smooth, the ideal reflecting case must be modified. The roughness of the sea surface will decrease the amplitude of the specular reflection by scattering some energy in directions other than the direction of the receiving platform. To account for this loss in received energy, a specular scattering coefficient, ρ_s, is used to modify the smooth-surface reflection coefficient ρ_o. Beckmann and Spizzichino[5] defined the mean-squared value of the specular scattering coefficient for a normally distributed isotropic rough surface as

$$\overline{\rho_s^2} = \exp\left(\frac{-4\pi\sigma_h \sin\psi}{\lambda}\right)^2 \tag{33}$$

where σ_h is the RMS wave height,
ψ is the incident signal's grazing angle,
and λ is the wavelength of the incident signal.

Thus, the mean specularly reflected power can be computed from the equation:

$$P_{Spec} = P_D \rho_o^2 \overline{\rho_s^2} \tag{34}$$

where P_D is the direct signal power,
ρ_o is the smooth-surface reflection coefficient,
and ρ_s is the specular scattering coefficient.

The computation of the coefficients in Equation (34) is a straightforward procedure. The smooth-surface reflection coefficient, ρ_o, is simply the magnitude of the of the complex reflection coefficient Γ, which was previously defined in Equation (5) as a function of the incident signal's grazing angle, ψ, and polarization characteristics, and the dielectric properties of the reflecting sea surface. Likewise, the mean square value of the specular scattering coefficent is determined from Equation (33). Representative values of ρ_s are plotted as a function of the roughness parameter g in Figure 3.

Before proceding to the discussion on the diffuse scattering component, it should be noted that the specular scattering coefficient defined by Beckmann and Spizzichino[5] relates to a surface-based radar and a low-altitude target. For the scenario of a semi-active homing missile engaging a target, a correction for the range difference between the direct and indirect signal paths must be applied. This correction is defined by the equation:

$$\rho_s' = \left(\frac{R_{MT}}{R_{MS} + R_{TS}} \right) \rho_s \tag{35}$$

in which R_{MT} is the missile-to-target range,
R_{MS} is the missile-to-specular point range,
and R_{TS} is the target-to-specular point range.

3.2 Diffuse Scattering Component

Contrary to the specular reflection phenomenon described above, diffuse scattering is a result of electromagnetic energy being reflected from a large area of the sea surface called the glistening area. Diffuse reflections have comparable magnitudes over the glistening area, but have completely random phases (incoherent). For modeling, these can be assumed to be uniformly distributed between 0 and 2π radians. The resultant magnitude of the diffuse scattering is Rayleigh distributed, with its magnitude density function expressed as

$$P(x) = \frac{2x}{P_o} \exp\left(\frac{-x^2}{P_o} \right), \quad for\ x > 0 \tag{36}$$

Model-B treats the diffuse scattering as an incoherent power source. As such, the mean diffuse scattered power component defined in Equation (32) can be written as:

$$P_{Diff} = P_D \rho_o^2 \overline{\rho_d^2} \tag{37}$$

where P_D is the direct signal power,
ρ_o is the smooth-surface reflection coefficient,
and ρ_d is the diffuse scattering coefficient.

The procedure used in model-B to compute the last term in Equation (37), the mean-squared value of the diffuse scattering coefficient, utilizes the theory developed by Beckmann and Spizzichino[5] and Barton.[7]

In order to obtain the total incoherent contribution of the glistening area, the component intensities of the electric field must be computed and summed together. These computations are based on the assumption that all of the diffuse scattering from the glistening area is a result of specular reflections from a matrix of rectangular sub-areas, or facets, which are used to represent the glistening area of the sea. The facets act a small mirrors in reflecting incident RF energy. By utilizing the *tangent-plane approximation*, in which the surface radii of curvature at all points on the surface are assumed to be considerably larger than the incident signal's wavelength, one can approximate the sea surface at and near any point as an infinite plane tangent to the surface at that point. This allows the relationship between the incident and reflected fields at the surface to be obtained from the principles of plane wave reflection from an infinite plane, or in this case the nonperfectly conducting sea surface.

If the wave height of the sea surface's glistening area is assumed to be normally distributed and isotropic in both surface dimensions, it was shown in Beckmann and Spizzichino[5] that the probability density function of β, the wave angle measure at a given point on the sea surface, is given by the equation:

$$G(\beta) = \frac{\exp\left(\frac{-\tan^2 \beta}{-\tan^2 \beta_o} \right)}{\tan^2 \beta_o \cos^4 \beta} \tag{38}$$

where $G(\beta)$ is the probability density function of β,
and β_o is the RMS wave slope angle.

The probability density function of β is also referred to as the bistatic scattering coefficient. Using this definition, the elemental diffuse spatial power density coefficient for a single surface facet was expressed by Barton[7] as the following:

$$d\rho_d^2 = \frac{R_{MT}^2}{4\pi R_{AM}^2 R_{TA}^2} \rho_o^2 G(\beta) F_d^2 \qquad (39)$$

where R_{MT} is the missile-to-target range,
R_{AM} is the missile-to-surface facet range,
R_{TA} is the target-to-surface facet range,
ρ_o is the smooth-surface reflection coefficient,
$G(\beta)$ is the bistatic scattering coefficient,
and F_d^2 is the surface roughness factor.

The two terms $\rho_o^2 G(\beta)$ are analogous to the *average incoherent bistatic scattering cross section per unit suface area* defined by Barrick.[6] For a linear vertically polarized RF waveform, the calculation of these terms is a straightforward procedure. However, for signals having other linear polarization characteristics, the computation of these terms involves a more complex procedure.

The surface roughness factor term, F_d^2, introduced by Barton,[7] represents the fraction of reflected power from a surface facet that contributes to the diffuse scattering. The roughness factor is computed from the equation:

$$F_d^2 = \sqrt{1 - \overline{\rho_s^2}} \qquad (40)$$

where ρ_s^2 is the mean-squared value of the specular reflection coefficient, as defined in Equation (33). This roughness factor is applied to both the incident and reflected signal paths, effectively using a geometric mean for the two grazing angles, as indicated by the expression

$$F_d^2 = \sqrt{\left(1 - \overline{\rho_{s1}^2}\right)\left(1 - \overline{\rho_{s2}^2}\right)} \qquad (41)$$

in which the subscript '1' denotes the incident signal and the subscript '2' denotes the reflected signal.

Therefore, by using Equation (39) to compute the elemental contribution of diffuse scattering from each facet, the mean square of the diffuse scattering coefficient is calculated as the sum of the elemental contributions from the entire surface facet matrix, or

$$\overline{\rho_d^2} = \sum d\rho_d^2 \qquad (42)$$

An extensive series of frequency domain computations are then performed for the RF signal waveforms scattered from the surface facet matrix to determine the center frequency and bandwidth characteristics of the diffuse reflection component. These calculations allow for the processing of the diffuse return in a separate high-fidelity receiver and signal processing model.

4. Model Comparison

Model-A is based on an earlier phenomenological vector model developed by Beard, Katz and Spetner[1] from empirical observations. Thus, the models used to represent both the coherent and incoherent scattering coefficients are based on this data, which is evident in Figures 3 and 4. One noteworthy point from these figures is the extrapolation of the observed data and how this feature has been modeled in the simulation. The downward trend modeled in the incoherent scattering coefficient for the larger values of the roughness parameter g is in keeping with Spetner's theoretical work.[9]

In contrast, model-B is based on the theory of scattering of electromagnetic waves from a rough surface as presented by Beckmann and Spizzichino,[5] Barrick,[6] and Barton.[7] The total reflected power is decomposed into mean specular and diffuse power components. The calculation of these mean power components is based on the Kirchoff approximation. For the specular component, this leads to a specular scattering coefficient having the general characteristics illustrated in Figure 2. However, as outlined in Section 3.1, a correction factor should be applied to this scattering coefficient when considering semi-active scenarios to account for signal path length differences. The diffuse components of the total power are computed by first subdividing the glistening area into a matrix of surface facets and then calculating an average incoherent bistatic scattering cross section per unit surface area for each facet. Unlike model-A, the calculation of the overall diffuse scattering coefficient in model-B is more tedious and requires integration over the surface glistening area.

In model-A, it was shown that the time correlation properties of the received reflected signal can be derived from the sea spectrum. On the basis of experimental observations, the Neumann model was adopted as a suitable wave height spectrum because of its bandpass characteristics. For fixed platform scenarios, the received signal spectrum can then be derived from the given wave height spectrum. A time series of the required multipath factors can then be rapidly generated using Gauss-Markov processes.

According to Northam,[4] this procedure yields results with second-order statistics which agree with experimental data. For moving platform scenarios, model-A uses the approach proposed by Northam.

On the other hand, frequency domain computations carried out using model-B are based on Doppler frequency calculation for each facet and the sorting of surface facet center frequency and bandwidth data from the entire surface facet matrix.

In summary, there are three major shortcomings of model-A[4]:

(1) The inability to accurately account for antenna pattern effects when the first Fresnel zone is not uniformly illuminated.
(2) The extrapolation of the scattering coefficients (both coherent and incoherent) to regimes where they have not been experimentally verified.
(3) The use of a model for the diffuse spectrum which has not been validated for moving platforms.

Likewise, the key limitations of model-B are:

(1) The rough surface scattering is based on the Kirchoff approximation.
(2) The effects of shadowing and multiple scattering are neglected.
(3) The numerical integration of the diffuse scattering coefficient and the Doppler processing of the diffuse power return is time consuming and computationally intensive.

The key features and limitations of model-A and model-B are summarized in Table 3.

5. Conclusions

In this paper, a functional description of two multipath models that are currently being used for predicting over-sea multipath effects in support of missile performance studies has been presented. Model-A, which has been implemented in the Weapons Systems Division at DSTO, is based on the phenomenological vector model of Beard[1-3] and the stochastic simulation approach by Northam.[4] Model-B, which is used in the Weapons Systems Department at NSWCDD, is based on the theory of scattering of electromagnetic waves from a rough surface as presented by Beckmann and Spizzichino,[5] Barrick,[6] and modified by Barton.[7]

A discussion of the key features of both models has been given and their major limitations identified. Due to space considerations, simulation results from each model were not included. A sampling of the preliminary results will be presented at the conference.

6. References

1. C.I. Beard, I. Katz and L.M. Spetner, "Phenomenological Vector Model of Microwave Reflection from the Ocean", IRE Trans., **AP-4**, April 1956, pp. 162-167.

2. C.I. Beard and I. Katz, "The Dependence of Microwave Radio Signal Spectra on Ocean Roughness and Wave Spectra", IRE Trans. on Ant. and Prop., **AP-5**, April 1957, pp. 183-191.

3. C.I. Beard, "Coherent and Incoherent Scattering of Microwaves from the Ocean", IRE Trans. on Ant. and Prop., **AP-9**, September 1961, pp. 470-483.

4. D.Y. Northam, "A Stochastic Simulation of Low Grazing Angle, Forward Scatter, Over-Water Multipath Effects", NRL Report 8568, Dec 1981.

5. P. Beckmann and A. Spizzichino, "The Scattering of Electromagnetic Waves from Rough Surfaces", Pergamon Press, 1963.

6. Barrick, D.E., "Chapter 9: Rough Surfaces," Radar Cross Section Handbook, Volume 2, G.T. Ruck ed., Plenum Press, New York, 1970, pp. 671-723.

7. D.K. Barton, "Low Angle Tracking", Proceedings IEEE, **Vol. 62**, No 6, June 1974, pp. 687-704.

8. B. Kinsman, "Windwaves; Their Generation, and Propagation on the Ocean Surface", Prentice-Hall, Englewood Cliffs, N.J., 1965.

9. L.M. Spetner, "Incoherent Scattering of Microwaves from the Ocean", Radio Science, **Vol. 10**, No. 6, June 1975, pp. 585-587.

	Model-A	Model-B
Approach	Phenomenological/Empirical	Theoretical
Key features	Vector addition, Time series, Simple	Incoherent power addition, Facet decomposition, Computationally intensive
Ocean surface parameters	RMS wave height Neumann Spectrum	RMS wave height, RMS wave slope
Assumptions	Reflected signal is sum of Specular and Diffuse vectors Horizontal/Vertical polarisation	Reflected signal is sum of Specular and Diffuse powers Horizontal/Vertical polarisation
Scattering coefficients	Specular/Diffuse coefficients based on Beard's experimental data	Specular/Diffuse power derived using Kirchoff approximation Total diffuse power calculated by summing over glistening area
Limitations	Parameters extrapolated for large roughness factors Inability to accurately account for antenna pattern effects when the first Fresnel zone not uniformly illuminated Model not validated for moving platforms	Kirchoff approximation, Shadowing and multiple Scattering effects are neglected.

Table 3. Comparison of Multipath Models

INDUSTRY INITIATIVE FOR REVISED TRAINING SIMULATOR VALIDATION PROCESS

by

Kendall W. Neville*
The Boeing Company
Seattle, Washington

Abstract

The increasing cost of producing data and modeling information which meet today's standards for qualification of high-level civil transport training simulators is a major concern for airlines and training centers. The cost of high-fidelity data packages is a direct result of the need to acquire and analyze large quantities of flight-test data and to ensure a close quantitative match of simulation results to airplane data for a wide variety of maneuvers and flight conditions. A proposal to revise regulatory policy for simulator qualification would allow the controlled use of the aircraft manufacturer's engineering simulation data to partially validate a training simulator which differs in an incremental, well-defined manner from a fully flight-validated simulator. Such differences include modeling of simple geometric changes to the airplane configuration or revisions to software in onboard computers. The Boeing Next Generation 737-600/-700/-800 program was identified as a test case to verify the proposed process and its application to simple body length derivatives. This paper discusses the successful cooperative effort by airplane manufacturers, airlines, simulator manufacturers, and regulatory authorities to develop a proposed alternative qualification procedure which satisfies the sometimes diverse objectives of the training simulator industry.

Introduction

Early in 1994, the Royal Aeronautical Society (RAeS) established an international Data Issues Working Group for civil transport training simulators. The Working Group consisted of representatives of airlines and other training providers, airplane manufacturers and other data providers, simulator manufacturers, and regulatory authorities. The purpose of the Working Group was to provide a forum for resolving matters relating to simulator data which were of principal concern to the industry. The full Working Group met three times from July, 1994, until April, 1995.

Several of the issues which were discussed by the Working Group related to the escalating cost of data packages for full-flight simulators. In his keynote address to the Working Group, Mr. Dieter Hass of Lufthansa German Airlines emphasized one potential remedy to this problem:

"The airplane manufacturers must get the cost of data production down."

Two closely-related proposals were presented to the Working Group which, if accepted, would help to achieve this objective.

---

* Senior Principal Engineer, Senior Member AIAA

Copyright © 1997 by The Boeing Company. Published by the American Institute of Aeronautics and Astronautics, Inc. with permission

The Initial Proposal

At the first meeting of the Data Issues Working Group in July, 1994, a total of ten issues were presented for discussion. Two issues which generated considerable interest and some controversy were proposals to revise regulatory approval policy for full-flight simulators. The revised policy would allow the controlled use of engineering simulation data for validation of full-flight simulators.

One proposal suggested that flight-test validation is not required for minor software changes in flight-control computers which are implemented after completion of the original flight-test program. The second proposal referred specifically to changes in fuselage length of one or more simple derivative airplane models.

The proponents of these proposals argued that full advantage should be taken of today's tools and methods to predict and identify actual airplane characteristics in order to avoid unnecessary flight testing and the associated labor-intensive analysis which is required for simulation update and validation. In addition to the obvious cost benefit, the proposed alternative process would enable earlier release of simulator data packages, and therefore earlier availability for flight crew training.

For the case of a simple geometric change such as the difference in body length for a derivative airplane model, the revised procedure would utilize proven predictive methods which have been developed by the major transport manufacturers from years of experience with wind-tunnel and flight-test measurements, as well as analytical methods which are based on a clear understanding of the effects of such changes. Validation data for a simple derivative simulation would primarily consist of data produced by the aircraft manufacturer's engineering simulator, supplemented, if necessary, by a limited set of flight data. Because the differences are incremental in nature and well-defined, such a derivative simulation still would be closely linked to the fully flight-updated simulation of the first model of the series[1].

After the second Working Group meeting, it became clear that these two issues were based on a common concept, and could be combined into one consolidated proposal:

An internationally accepted policy should be defined which would allow the controlled use of aircraft manufacturer's engineering simulation data for training simulator validation in lieu of flight data for small, well-defined changes from a fully flight-validated simulation.

While the Working Group recognized the potential cost and schedule benefits of this proposal, some members expressed concern that such a change in simulator approval policy could compromise the established requirement for a quantitative comparison of simulator results with actual flight-test data measured from the simulated airplane. Simulator qualification standards such as those published by the FAA[2], the JAA[3], and ICAO[4] state that the airplane manufacturer's flight-test data are the preferred source of validation for initial simulator qualification.

These and other concerns were further discussed at the second meeting of the Data Issues Working Group in September, 1994, and at the third meeting in April, 1995.

The Resolution

Much of the third two-day Working Group meeting was devoted to the proposal to allow the use of engineering simulation data for simulator validation. After lengthy discussion and several side meetings, the following resolution was accepted by the full working group:

This Working Group believes that:

In the case of a flight-validated simulation which is modified to represent changes to the airplane configuration and/or a system, use of the aircraft manufacturer's modified engineering simulation data may be an alternative to obtaining flight test data for validation of training simulators, subject to:

a. *A generic process being established to ensure consistent implementation, control and regulatory approval of this approach.*

b. *A task force being established to define this process.*

c. *A prototype activity being established and evaluated, as an initial phase, to assess validity of this approach.*

d. *A resolution package being prepared for approval by this Working Group for presentation to an RAeS Conference.*

The Task Force

In accordance with the Working Group Resolution, a dedicated Task Force was formed at the meeting of April, 1995. The Task Force consisted of the following representation from the flight-crew training industry:

- Airplane manufacturers: Airbus, Boeing, and McDonnell Douglas
- Airline training departments: United, Lufthansa, USAir, SAS, and Swissair
- Regulatory agencies: United States, United Kingdom (The representatives of SAS and Swissair also represented their respective national aviation authorities.)

The simulator manufacturers agreed that their participation with the Task Force was not required.

The following Task Force objectives were established:

- Define the scope of acceptable changes to a baseline airplane model for which engineering simulation data may be used in lieu of flight data for the purpose of training simulator validation.
- Establish configuration management and regulatory evaluation policies for the aircraft manufacturer's engineering simulator.
- Develop an implementation process for approving the use of engineering simulation data for training simulator validation.
- Identify a test case to use as proof of concept. It was agreed that the Boeing 737-600/-700/-800 program, a simple body-length derivative family scheduled for flight-testing in the 1997-98 time period, was well-suited for this purpose.

It was recognized by the Working Group that the successful implementation of the proposed revision to simulator approval policy would require concurrence from the worldwide regulatory community. The basis for today's standards for flight simulator qualification is the ICAO Manual for simulator qualification[4]. This reference was developed by an industry-wide effort which included the active participation of the international regulatory community. Similar international cooperation would be required to allow the limited use of engineering simulation data for training simulator validation.

To obtain such international acceptance, a comprehensive presentation would be prepared by the Task Force explaining the

benefits to the industry of the proposed revised policy and address all known and anticipated concerns. In addition, supporting technical data from current airplane programs would be presented by each of the three major airplane manufacturers.

Finally, the Task Force would prepare a detailed proposal for presentation to an RAeS flight-simulation conference with a request for the endorsement of the conference delegation.

A summary of key events for the Task Force activity from its formation in April, 1995, until the RAeS conference in May, 1996, is shown in Figure 1.

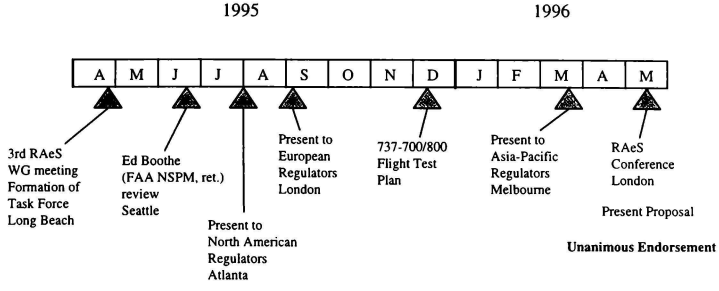


Figure 1 Key Events for the RAeS Engineering Simulator Data Task Force Activity

Presentations to Regulatory Authorities

Prior to developing the material for presentation to the international regulatory community, members of the Task Force met with Edward M. Boothe, former FAA National Simulator Program Manager. Mr. Boothe was instrumental in the development of the FAA Advanced Simulation Plan which serves as the basis for today's standards for validation and qualification of full-flight simulators. Mr. Boothe, now an independent consultant, provided valuable insight into the probable concerns that would have to be addressed, and strongly influenced the content of the presentation. Mr. Boothe was very supportive of the concept, and felt that if accompanied by the proper conditions, it would be entirely compatible with the Advance Simulation Plan.

The detailed presentation was then prepared, and one-day meetings were scheduled with regulatory authorities in each of three major geographical areas: North America, Europe, and the Asia-Pacific region.

The first meeting was held with the FAA and Transport Canada in Atlanta, Georgia, in August, 1995. A frank and constructive discussion followed the presentation during which recommendations for improving the proposal and its implementation were offered. The plan was approved by the authorities, subject to some modifications based on the discussion comments.

The second presentation was given to the European Joint Aviation Authorities JAR-STD Working Group in Gatwick, England in September, 1995. The final meeting was held with Asia-Pacific authorities in Melbourne, Australia in March, 1996.

Following each meeting with regulatory authorities, the RAeS Task Force met to discuss and evaluate the comments and adjust the plan and presentation content as necessary.

Regulatory Concerns

Several concerns were frequently expressed during the discussion period which followed each presentation. It was recognized by the Task Force that to achieve the approval of the regulatory community for the proposed alternative qualification procedure, each of the following conditions would have to be satisfactorily addressed:

- Can't allow simulation fidelity to decrease from current standards.
- Can't retreat from the FAA Advanced Simulation Plan requiring a flight-test basis for validation.
- Can't depart too far from a flight-validated simulation.
- Must limit the revised process to qualified applicants, such as the large airplane manufacturers.
- Must define requirements for the aircraft manufacturer's engineering simulator.
- Must define the allowable types of changes relative to the flight-validated model.
- Must provide sufficient results from a test case using a future flight-test program to adequately validate the process.

All of these concerns were carefully considered in updated versions of the presentation, and in the detailed draft proposal which was being prepared for presentation to the RAeS flight simulation conference.

The Detailed Proposal

The detailed draft proposal[5] represents the views of the Data Issues Working Group and the consensus of the international regulatory community. The proposal consists primarily of two sections:

1. A main body which contains background information, a discussion of the present situation, and the plan developed by the RAeS Task Force. The 737-600/-700/-800 program is identified as the test case, or "prototype activity", to validate the concept and its particular application to simulations characterized by simple geometric changes from a fully flight-validated model.
2. An appendix which provides "draft regulatory material" which could be inserted directly into the JAR-STD simulator qualification guidance document[3]. The proposed new section of the JAR-STD document, entitled, "Engineering Simulator as an Alternate Source of Validation Data", explains the requirements that must be met to qualify for the proposed procedure.

The draft regulatory material addresses each of the major concerns which were expressed by the regulatory authorities. Of fundamental importance are criteria which would ensure that there be no degradation in fidelity for simulator models which qualify for the proposed alternative process. Applicants must have a proven record of high-fidelity simulation, and must have

demonstrated high-quality prediction methods through comparisons of predicted and flight-test data. As further assurance that fidelity would not be compromised, application of the process would be limited to "one step away" from a fully flight-validated simulation. Such a "one-step" difference could include two or more related changes, e.g., a body length change plus software revisions which are required to preserve similar handling characteristics.

In response to the request for a clear definition of requirements for the airplane manufacturer's engineering simulator, the draft JAR-STD material sets the following standards which would have to be met by such a facility:

- *Have to exist as a physical entity, complete with a flight deck representative of the affected class of airplane, with controls sufficient for manual flight.*
- *Have a visual system, and preferably also a motion system*
- *Where appropriate, have actual avionics boxes interchangeable with the equivalent software simulation, to support validation of released software.*
- *Have a rigorous configuration control system covering hardware and software.*
- *Have been found to be a high fidelity representation of the airplane by the manufacturer's pilots and by the regulatory/operator's pilots.*

Again in response to regulatory concerns, the draft proposal restricts the types of modification covered by the alternative procedure to those with "well-understood effects", specifically:

- *System software*
- *Simple geometric revisions (e.g. body length)*
- *Engines (limited to non-propeller-driven airplane)*
- *Control system gearing, or deflection limits*
- *Brake, tire and steering revisions*

Finally, the draft proposal was presented to the Royal Aeronautical Society flight simulation conference on May 17, 1996. The format for the presentation included perspectives by regulatory authorities, airplane manufacturers, airline training departments, and simulator manufacturers, all in support of the proposal.

Following the conclusion of the presentations, the proposal was unanimously approved by the conference delegation.

The Prototype Activity

The proof-of-concept, or "prototype activity", using the Boeing Next Generation 737 flight test program, will serve a twofold purpose:

1. Provide validation of the proposed alternative process for the use of engineering simulation data, especially for simple derivative models.
2. Validate Boeing prediction methods for the effects of body length change.

The Next Generation 737-600/-700/-800 family is a derivative of the current 737-300/-400/-500 family. The 737-700 model, the first of the series, is a major derivative of the 737-300 airplane. Although its fuselage is the same size as the 737-300, the 737-700 has a new wing and high-lift system, and larger horizontal and vertical stabilizers. The 737-800 and 737-600

models are longer and shorter derivative versions, respectively, of the 737-700, with only minor system changes which are directly related to the differences in body length. Superimposed plan view drawings of the three Next Generation 737 models are shown in Figure 2.

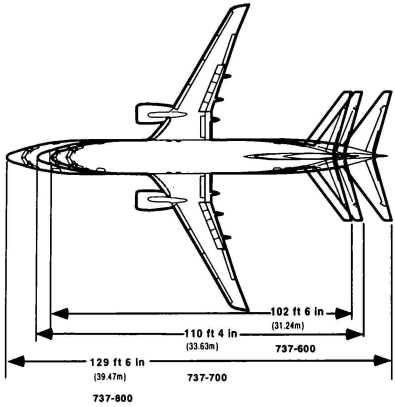


Figure 2 737-600/-700/-800 Configuration Comparison

The flight-test program to certify the Next Generation 737 series began in February, 1997, with the first flight of the 737-700 model. The 737-800 and 737-600 are scheduled to begin flight testing in late July, 1997, and February, 1998, respectively.

The mid-size 737-700 simulation will be fully flight updated and validated using data collected during its flight-test certification program. In accordance with the RAeS Working Group proposal, and as approved by the regulatory authorities, a reduced set of flight data will be performed on the longer 737-800 model to support the proof-of-concept activity. This reduced set will consist of about one-third of those tests which are specified by the regulatory guidance material[2,3,4] for qualification of full-flight simulators. These tests include conditions which are known to be significantly affected by body length, as well as a few which are intended to confirm that some maneuvers are virtually unaffected. Some tests may identify areas where current prediction methods are inadequate to represent actual airplane behavior for a body-length derivative. For the latter case, some flight testing may be required for simulator validation of subsequent simple derivative programs, such as the 737-600.

The following sequence of events was established by the Task Force for the prototype activity, in accordance with the guidelines of the draft proposal:

- An application was submitted to the regulatory authorities in January, 1997.

- Meetings with key regulatory authorities were held to discuss and request approval for the detailed plan.
- The 737-700 simulation will be flight updated.
- The 737-800 simulation will be updated using only the 737-700 flight-updated simulation as a baseline, then applying established methods for predicting the incremental effects of body length.
- The simulator-specific flight-test data for the 737-800 will then be collected.
- Flight test results will be compared to the updated 737-800 simulation and submitted to an appropriate Working Group of regulatory authorities.
- Finally, results of predicted-to-flight comparisons will be used to determine which flight-test maneuvers (if any) will be required for validation of the short-body 737-600 simulation.

The key events for the proof-of-concept activity are summarized in Figure 3. The darkened symbols indicate those events which have been accomplished at the time of this writing.

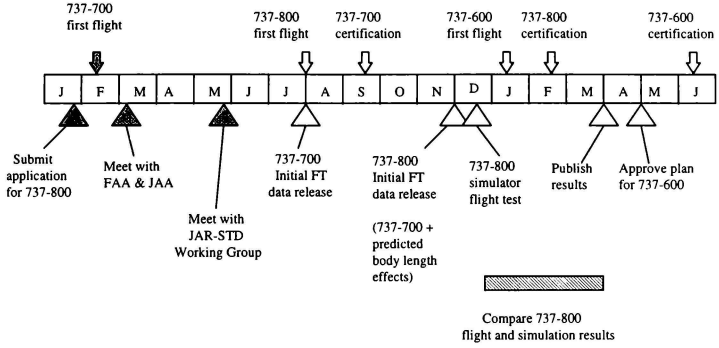


Figure 3 Key events - The Proof-of-Concept

Implementation of the Revised Process

The RAeS Working Group draft Proposal[5] states that if the 737-600/-700/-800 proof-of-concept activity is successful, the outcome of this review process should be a recommended amendment to the regulatory documentation and to the ICAO manual[4] which defines this alternative means of providing validation data for high-level full-flight simulators.

The IATA simulator requirements document[6], the established reference for defining the standards and scope of data required for full-flight simulators, now includes a paragraph which supports implementation of the proposed revised process. The 5th edition of the IATA document, published in 1996, states:

In the case of a fully flight-test validated simulation which is modified as a result of changes to the airplane configuration or systems, the airplane manufacturer may supply validation data from an audited engineering simulator to supplement flight-test data, subject to the prior agreement of the regulatory authorities.

Conclusion

A Data Issues Working Group was established by the Royal Aeronautical Society to resolve issues relating to the cost, quality, and support of data for flight-crew training simulators. One issue which generated considerable interest and some controversy was the proposal to allow the controlled use of engineering simulation data for validation of simulators characterized by well-defined incremental changes from a fully flight-validated simulation. Close industry cooperation resulted in a proposed implementation procedure which would balance the requirements for today's high standards for simulation fidelity with the need for prudent measures to control the cost of data production. The willingness of all participants to consider what is best for the industry as a whole has been the key to the success of this activity. The objective of including the revised process in future regulatory documentation depends on the successful outcome of the current proof-of-concept activity involving the Boeing 737-600/700/800 flight-test program.

References

1. RAeS paper, "More Cost Effective Method for Update and Validation of Simple Derivative Simulations", K. W. Neville, 1996

2. Federal Aviation Administration Advisory Circular AC120-40B, "Airplane Simulator Qualification", July 29, 1991

3. Joint Aviation Authorities NPA-1, "JAA Requirements for Flight Simulators", (draft), December 17, 1994

4. International Civil Aviation Organization, "Manual of Criteria for the Qualification of Flight Simulators", first edition, 1995

5. RAeS Data Issues Working Group Engineering Simulator Data Task Force, "Proposal for Use of Engineering Simulator Data as a Supplement to Flight Test Data to Support Flight Simulator Qualification for Modified or Derivative Aeroplanes" (draft), May, 1996

6. International Air Transport Association document, "Flight Simulator Design & Performance Data Requirements", 5th Edition, 1996

AIAA-97-3666

VALIDATING NASA'S "SURFACE DEVELOPMENT AND TEST FACILITY" AIR TRAFFIC CONTROL TOWER SIMULATOR

Richard F. Haines, Ph.D.
Sr. Research Scientist, RECOM Technology Inc., Ames Research Center-NASA
Moffett Field, Calif. 94035

Abstract

The proper presentation and management of information in America's largest and busiest (Level V) air traffic control towers calls for an in-depth understanding of many different human-computer considerations: user interface design principles for graphical, radar, and text information; manual and automated data input hardware; information/display technology; reconfigurable workstations; workload assessment; and a comprehensive understanding of air traffic control procedures that are currently in use. This paper discusses these subjects in the context of validating the *Surface Development and Test Facility* (SDTF) a full scale, multi-manned, air traffic control research simulator currently under construction at NASA's Ames Research Center. Successful validation must demonstrate that human and system performance in the SDTF is equivalent to that in the actual airport control tower.

Introduction

Airport tower facilities across the country will have to adapt to accommodate unprecidented levels of air traffic growth that are predicted over the next several decades and the higher traffic capacities made possible by the implementation of air traffic control (ATC) automation, such as the Center "Terminal Radar Approach Control" (TRACON) Automation System (CTAS) [which includes such tools as the Traffic Management Advisor (TMA), Expedite Departure Path (EDP), and Final Approach Spacing Tool (FAST)] and the Surface Movement Advisor (SMA).

In order to accommodate the operational changes brought about by increased automation, it is important to investigate some issues, through simulation, prior to fielding these systems. For example, do tower procedures need to be modified in order to maximize system efficiency using the Expedite Departure Path (EDP) tool? How can tools like SMA be used most effectively in day-to-day operational contexts? Can training time be reduced by standardizing tower facilities? How can ATC towers work more effectively with airline operation centers to reduce congestion and expedite airport surface traffic flow? We face these and many other such questions as we enter the twenty-first century.

"This paper is declared a work of the U.S. Government and is not subject to copyright protection in the United States

Surface Development and Test Facility Overview

The Surface Development and Test Facility will be a full scale, fully-staffed, air traffic control simulator that provides the "look and feel" of an actual Level V airport tower cab (e.g., Atlanta, Chicago, Denver, Los Angeles, New York). It is now under construction at Ames Research Center - NASA, Moffett Field, California.

This two story facility will have rooms on the first floor for pseudo-ramp controllers, simulation engineers and investigators, test debriefings, and a computer room. In addition, there will be a room for pseudo-pilots who will generate realistic air traffic input for the tower cab controllers working on the second floor. The second floor tower cab work area (31 feet across) will be surrounded (360 degrees) by twelve, large, sloped windows. Twelve edge-to-edge (rear) projection screens measuring 120" wide by 90" high are located outside these windows. The continuous 360 degree external environment will display highly realistic day- and night-time scenes and weather conditions. (see Figure 1) Up to 200 moving vehicles may be displayed at any moment. SGI Onyx Infinite Reality graphics engines and 3-D graphics software will draw these scenes. SDTF furniture will permit reconfiguration to represent any Level V tower in America. Other details of its various sub-systems are provided elsewhere (http://sdtf.arc.nasa.gov).

The SDTF differs from other existing 360 degree simulators in that it is designed to support research, it is

Figure 1

Computer Rendering of Completed
Tower Cab Environment



highly flexible in its ability to be reconfigured, it is relatively large and able to accommodate a full complement of staff, and it is highly realistic both visually and auditorily. It will provide a needed bridge between advanced research, functional testing, validation, and final deployment to the field of new procedures, hardware, software, and operational concepts for the air traffic control towers of the future. It will also support cost-benefit studies, provide a stable platform from which new requirements can be derived, enable information sharing to occur among multiple users, and test software performance, safety, and reliability under realistic conditions.

The SDTF will provide an environment to test the performance, safety, and reliability of prototype software under various airport configurations and to evaluate new computer-human interface hardware displaying integrated information from multiple data sources to various users. For example, algorithms to monitor and model low probability-of-occurrence errors, and time-and-motion studies, etc. can be evaluated. It is anticipated that these pre-field committment evaluations will improve overall safety. Also, tower-related incidents can be simulated, replayed multiple times and analyzed if necessary.

Need for Facility Level Validation

Need for Validation Guidelines. What are "validation guidelines" and why are they needed? They are clearly defined approaches to testing the operational efficiency and accuracy both of individual sub-systems within the SDTF as well as the complete facility to ensure that it meets all desired objectives. Because humans will load software, operate equipment, view displays, make complex decisions, and interact with other people in the SDTF, a human factors-oriented test plan is needed. This paper recognizes many of the human-machine issues along with system maintainability, fault detection - diagnosis, and other personnel-related issues which likely would go untested if the hardware and software were tested alone.

Use of these validation guidelines will help to ensure that the research conducted in the SDTF will take into account all relevant functional requirements so that it will answer basic questions of simulation fidelity and experimental soundness.

Need for Facility-wide Operational Flexibility. The SDTF will be used by a number of different user groups including: air traffic controllers, airline operators, airport operators, pilots, scientists, and test engineers. Since it will serve the nation as a research simulator it must be able to faithfully represent not only existing American Level V towers but also future control cabs which may incorporate very highly advanced technology. These constantly changing needs will place a heavy burden on the SDTF and its support staff in terms of its capability to changeout: software and hardware, displays and controls, small to large user teams, and partially- to fully-automatic operational procedures. Scheduling of the different classes of SDTF users alone will call for new and innovative approaches to facility organization. In short, operational flexibility will become a key factor in the future success of the SDTF.

This need for total system flexibility must go beyond merely including the correct "hooks" and "scars" into the building's initial design. Providing for adequate operational flexibility also calls for significant insight into where the trends in information (generation, storage, management, display) are likely to move in the future and what kinds of research and development may be conducted in the SDTF in years to come.

SDTF validation will be necessary in order to ensure that not only do all displays, sensors, algorithms, and controls function correctly but that operator behavior and performance in the simulator, i.e., human factors related responses, are fundamentally equivalent to those found in an actual tower facility. Once validated, the insights and hard data obtained in the SDTF will map back upon the real world with sufficient accuracy to permit safe data extrapolations to be made. Currently, efforts are underway to insure that ISO 9000 standards form an integral part of the day-to-day operation of the facility.

General SDTF Validation Principles

It is important to state some initial general principles the acceptance of which may help to achieve a consistent and workable program of research and development over the years. They include:

1. Involve Highly Experienced Users. Validation should involve users who are already fully familiar with the system(s) being tested. Experts are in the best position to assess simulation accuracy.

2. Develop a Knowledge Database from Actual Towers. Operations at actual towers will be observed and documented with particular attention to procedures, staffing, communications, equipment layout, etc. Data in the Aviation Safety Reporting System and Aircraft Communications, Addressing, and Reporting System (ACARS) reports will also be researched to identify possible problem areas related to ATC operations.

3. Small to Large. Validation should proceed from small to large, from sub-system to full system whenever possible in order to identify problems and malfunctions which otherwise might be missed if embedded within a larger environment.

4. Individual Airport Revalidation. Validation should be repeated with the installation of every new airport database.

5. Begin with Familiar Displays. Validation should involve the use of displays and controls with which the staff are already familiar and only later (when required) introduce new displays and controls. From a research methodology standpoint, this approach will make it much easier to interpret test findings.

6. Facility Validation not Training or Research. While SDTF validation may involve training of NASA and support contractor personnel on new tasks and equipment as well as research to evaluate their performance under some circumstances, facility validation should take first priority.

7. Define all Measures Clearly. Validation should, whenever possible, include clearly defined, quantitative measures which can be related back to the real-world, e.g., measures and parameters from a particular airport which already has existing operational statistics.

8. Validate after Sub-systems Check Out. Validation should begin only after appropriate SDTF personnel have completed all required calibration procedures and have certified the accuracy of these components, e.g., the outside environment scene generation system must be optically calibrated, aligned, color converged, correctly size-scaled, etc. beforE using it to display air and ground traffic.

9. Configuration Control Validation Issues. Configuration control must include systematic documentation of both initial (i.e., as installed) sub-systems as well as all subsequent upgrades.

10. Aeronautical Safety is Paramount. Whenever feasible, validation testing should emphasize aviation safety over everything else. If there is a conflict between achieving operational safety in the simulated aeronautical environment and conducting other research, safety should predominate.

The SDTF validation process will proceed along two parallel paths, engineering validation and human factors validation. The former deals with various aspects of simulator performance at the component and the system level. System level validation procedures will include (but are not limited to):

- Accuracy of aircraft performance/flight path
- Conformance of the SDTF with established Federal Aviation Administration (FAA) certification standards
- System response time(s)
- Interior/exterior illumination levels
- External scene display characteristic(s) realism
- Auditory cue(s) fidelity

The remainder of this paper deals with selected aspects of human factors validation of the SDTF.

Human Factors Guidelines for Facility Validation

The field of human factors engineering has progressed well beyond the field of ergonomically based equipment design. It has matured significantly over the years to the point of being able to offer many useful techniques and guidelines with which a complex, multimanned facility, like the SDTF, may be evaluated (Haines, 1990a; Tillman and Tillman, 1991; Wise et al., 1993). These techniques range from simple observational task analyses (in which each person in the work area is carefully monitored to find out exactly what they do and when) to microrecording and analyses of user-system interaction during which individual input (e.g., key strokes to a computer) are related to some particular desired outcome at a later time. (cf., Boff and Lincoln, 1988; Haines, 1990b) Helander, 1988; Woodson, et al., 1992).

Another potentially valuable insight concerning the nature of human-centered systems is that they may be

viewed from three different perspectives: usability, suitability, and user acceptability (cf., Harwood, 1993).

Usability refers to various perceptual and psychophysical characteristics of a facility. Suitability reflects the capability of the system to provide the required functionality and operational information. Final acceptability of the facility is (usually) dependent both upon its ability to meet the first two issues as well as personal concerns like job satisfaction, etc. However, whether the SDTF is realistic and whether its level of realism is acceptable are two quite different but important questions which should be assessed independently.

To a large extent, valid generalization of SDTF research results will depend on achieving a sufficiently close match between the simulated ATC environment and the real world. Thus, validation is an essential step towards the eventual acceptance of the SDTF by its intended users.

Following are five human factors guidelines that follow from the ten general approaches to validation, given above, which should be followed while validating subsystems or fully integrated systems within the SDTF.

Guideline One. **Follow current Air Traffic Control procedures as a "baseline" control condition.** In order to permit both valid and maximally applicable assessments to be made concerning a new operational procedure it is necessary to have a behavioral data baseline against which its performance may be judged (McGuigan, 1968). Pre-existing, clearly documented ATC procedures should serve as this "control" condition whenever possible. These baseline procedures should be identified prior to testing in the SDTF. If these validation activities show that current controllers can carry out their normal duties in all respects we can be more confident that the SDTF will be able to support future high fidelity simulations to address theoretical research questions. In short, real-life procedures should define the early validation tasks.

Another benefit to be gained from following current ATC procedures is that staffing requirements can be assessed more accurately. For example, can the staff readily change positions and also communicate with each other in the SDTF as they would in the real tower?

Guideline Two. **Identify the user- and system-related test and efficiency metrics in advance of any evaluation or study.** The importance of having a set of written evaluative criteria or metrics prior to designing and conducting a study cannot be overemphasized. They will not only help avoid many poorly recognized problems but will illuminate potential solutions to new ones.

Guideline Three. **Use existing displays and controls as a "baseline" control.** Does the SDTF equipment behave in the same way as in the real tower cab? Can all required tasks be performed in the SDTF? Does the equipment provide the same look, feel, feedback, delay, and unreliability as is found in the actual environment? If some new display and/or control device is being considered for installation in the SDTF then conduct a "A-B; B-A" type study where "A" represents the pre-existing display and/or control device and "B" represents the new one. Each controller should be requested to use "A" for a period of time followed by "B" (for the same duration). After a suitable rest period the same controller is requested to use "B" first followed by "A" for the same duration. This "balanced" experimental approach will help control for fatigue effects, learning effects, and other evaluation problems. Suitable evaluation ratings and workload assessments may be administered during or immediately following each of these work sessions.

Guideline Four. **Hold the work environment as constant as possible during all evaluations.** If, for instance, an evaluation of a new radar (D-BRITE) display symbology is to be evaluated, and guidelines 1, 2, and 3 (above) have been followed, also hold ambient illumination and auditory levels at a constant, realistic level throughout the evaluation period.

Guideline Five. **Organizational equilibrium in the real world does not always need to preceed new technology research in the SDTF.** The old adage "if it isn't broken don't fix it" needs to be tempered by a another - "if it isn't broken it may break at any time." For example, while ATC organizational stability within a given tower may be seen as a good sign, indeed an indication that all is well, a working environment having such apparent equilibrium may not be able to cope effectively with certain unanticipated (i.e., sudden, unfamiliar) events in the aviation environment. All organizations have their limits of ability to cope with new situations. So SDTF management should be constantly seeking new solutions to existing situations as well as to impending problems.

List of Candidate SDTF Human Factors Validation and Research Areas

Human factors validation will employ both objective and subjective methods in order to insure that the real operational environment is realistically simulated. Objective measures will include: (a) Comparison of controller decisions and resulting traffic flow

during actual traffic conditions with the decisions and results from simulated traffic periods. (b) Measurement of pseudo-pilot and pseudo-ramp controller response times and error rate(s) to simulated yet realistic control tasks. (c) Determination of the number of aircraft worked, how often certain displays are accessed, and personnel communication links used and analyzed. (d) The number, duration, and content of communications among all controllers and between controllers and their supervisors may be quantified. Such communications also may be compared with similar measures obtained from the actual airport tower. Thus, selected pre-approved evaluations may need to be conducted at current Level V air traffic control towers to provide an understanding of "nominal SDTF baseline" performance.

The objective of these manned simulation tests is to clearly demonstrate that the SDTF provides a highly realistic, yet reconfigurable work environment which duplicates the visual, auditory, ergonomic, and cognitive environments of existing Level V towers and that real-time, multi-manned research can be fully supported. Aircraft arrival scenarios ("built" from radar tapes) and departure lists that are based on actual schedules filed during the same period will be generated to help ensure that realistic traffic flows and densities are achieved. Figure 2 illustrates one such display for an airport having parallel runways. A wide variety of human factors research may be carried out on touch-screen-activated screens such as this one where the controller has ready access to actual and predicted air traffic over event horizons which the controller can define.

Figure 3 shows a prototype airport ramp controller's computer display of approach (BLOCK IN) and departure (BLOCK OUT) information which may be used in the pseudo-ramp controllers' room of the SDTF. Since the airport represented here possesses parallel runways there are two direction (north and south) arrivals; (ACID = aircraft identification; GT = gate number; TIME = local time; S = status). Various human factors assessments may be made on display lists such as this one including font (brightness, color size, shape) visibility, information field layout taking into account visual eye scan dynamics, use of flashing or "strobed" symbols, optimal information update techniques, etc. Once these scenarios are completed current controllers will be brought to Ames Research Center to "run" each scenario under controlled and carefully monitored conditions.

Not to be overlooked is the SDTF audio system's performance. It must provide realistic and flexibly controlled voice communications with understandability that is equivalent to that found in an actual tower.

Since objective data alone is not expected to provide an adequate validation measure, subjective data will be obtained from the controllers. Because they are expert in the use and interpretation of the tower facilities, the controller's responses will be given considerable weight. It should be apparent to them where inconsistencies or inadequacies exist in the simulation.

Subjective data will be collected in several formats. Structured debriefing interviews, useability questionnaires, and workload ratings will be employed to augment the objective data. Such techniques can be used at the component as well as the system level.

Periodic follow-on validation activities will take place at both the component and systems levels and will employ both objective and subjective testing. It is likely that certain pre-test validation activities will precede the introduction of each new tower to be simulated in the SDTF. Computer input data may consist of both actual radar data feeds from local TRACON facilities as well as pseudo-pilot inputs, pseudo-ramp controller inputs, specialized experimenter data flagging inputs, and others. These validation procedures should contribute significantly to operating a world class simulator that will enhance America's flying safety.

Specific Candidate Validation Tests

Validation will be performed in three stages: 1. Initial facility component and full system validation, 2. Human factors validation, and 3. Periodic (later) validation. Upon completion of SDTF construction all installed *components* will be evaluated using appropriate objective criteria and methods, e.g., analysis of real-time, TRACON radar data feed throughput, external scene generation delays and update rates relative to the real world, optical alignment (color convergence, distortion, edge matching) of the twelve image projectors, telecommunications bandwidth measurement and signal handshaking, etc. The *integrated systems* will be tested during initial facility set-up using both objective and subjective methods. They will be performed on an airport-by-airport basis, i.e., if the Atlanta airport is selected, operating parameters from that particular facility will be used to compare against the SDTF's simulation of Atlanta, etc. Various individual tower physical layout conformance measures such as furniture layout, traffic flow, sight-lines, etc. will be evaluated as well.

Because the SDTF is very complex it could be studied using literally dozens of different research methods. Table 1 lists some other candidate research areas to evaluate SDTF sub-systems as well as the entire facility from a human factors point of view:

Table 1

Candidate SDTF Human Factors
Validation and Research Areas

Automation Development and Evaluation

New ATC technology transfer. Procedures are used to compare existing with new ATC technology on one or more evaluative dimensions.

New Graphic User Interface (GUI) development. Systematic comparison of the acceptability, understandability, usefulness, and general utility of new user (information) interface(s).

Software comprehensibility. The ability to understand what a program does and how it does it. (cf., Helander, 1988, pp. 107- 121)

Software compatibility. The ease with which new software can be integrated into existing software.

Introduction of innovation. Determinations of how and when to optimally introduce innovative ideas and products into an existing tower environment.

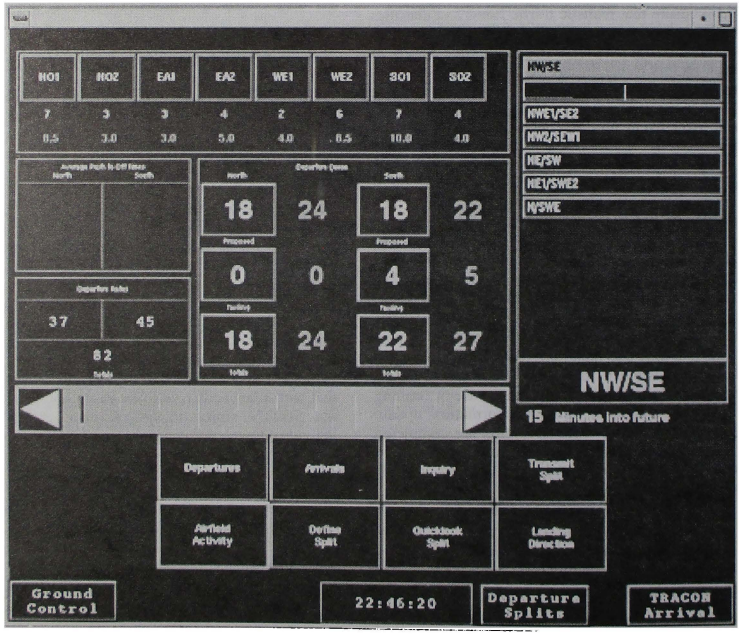
System Evaluation

Performance error isolation. Identification of low-probability performance errors which could compound into actual incidents or accidents and how to effectively mitigate them in advance.

User workload issues. Tests to determine mental, physical, and psychological workload generated by a given work situation.

Figure 2

Sample Information Display Screen
Used in Advanced Tower Cab Research



Cognitive Issues

Controller decision support aids. Special techniques used to find out if computer-based information really contributes to making better (more accurate and more timely) decisions in real time? Are operational ambiguities really reduced?

Model development. New procedures are tested to see if they contribute greater understanding to new behavioral or cognitive-engineering models. Predictive models of human performance may be developed.

General problem solving. A practical problem is simulated and performance on each task is monitored and analyzed to understand the factors contributing to its solution. System fault detection - diagnosis studies should be conducted.

Miscellaneous issues. Tests of group dynamics and functional interrelationships occuring over typical work periods conducted to identify optimal work factors.

In general, there are three factors which must be carefully considered in attempting to introduce new technology into existing tower cabs: (1) the specific features of the new technology which may require operator training and certification, (2) the defining characteristics of the organizational context in which the new technology is being introduced, and (3) the way that the implementation process is carried out. Each of these factors can play a crucial role in determining the success of the proposed technology transfer.

To the extent that each Level V tower represents a unique environment it is neither wise nor safe to assume that what works in one tower will work in another. In short, the existing organizational procedures used in each facility must be taken into account in planning future SDTF validation tests. Tower operators from a particular level V tower should be employed in a SDTF validation test of that airport.

The span of possible research issues is very broad. The SDTF must be flexible enough to accommodate these diverse research objectives. In addition, the SDTF

Figure 3

Sample Ramp Tower Controller Screen

RAMP SCREEN 3

North Arrivals				Departures					South Arrivals			
BLOCK IN				BLOCK OUT					BLOCK IN			
ACID	GT	TIME	S	ACID	GT	DA	TIME	S	ACID	GT	TIME	S
DAL419	B15	1744	L	DAL1644	B23	N2	1751	D	ASE117	C28	1753	D
DAL422	B25	1723	L	DAL2055	B05	N1	1749	D				
DAL1041	B11	1800	M	ASE539	C30	W1	1744	D				
DAL396	B23	1758	H	ASE284	C26	S2	1748	D				
DAL453	B21	1757	M	ASE259	C32	S1	1741	D				
				DAL610	B13	S2	1752	T				
				DAL1572	B07	E2	1751	T				
				DAL1203	B31	N2	1750	T				
				DAL638	B15	E2	1743	T				
				DAL1694	B11	E2	1516	T				
				DAL422	B25	N2	1751	PP				
				ASE214	C24	S1	0526	S				
				ASE845	C26	W1	0526	S				
				ASE63	C24	N2	0527	S				
				ASE46	C24	E2	0527	S				

NW/SE WEST 17:53:13 Main Swap

must integrate well with existing ATC software and hardware and must also accommodate prototypes associated with ATC modernization efforts.

Validation as an Ongoing Activity

SDTF use will be contingent on two different sets of issues: system performance and facility conformance. The former should be fairly uniform across simulations of different Level V towers. The latter, on the other hand, must take into account airport-to-airport differences (information and control interfaces, furniture, controller movement traffic and communication patterns, aircraft traffic scenarios, idiosyncrasies of weather) and other factors. As facility procedures and airline schedules are constantly changing, this level of validation will be ongoing throughout the course of SDTF use.

Some Proposed SDTF Validation Tests

This section suggests several possible strawman tests for the SDTF which are typical of actual airport tower operations. Each represents a validation exercise which can include a research dimension as well. Of course they represent only a small sample of such tests.

1. Pushback Rescheduling. In this test scenario ground controllers will use the SDTF to play a "What if" game which involves delaying push-backs from the gate by varying durations when they feel that a larger gain in scheduling is likely to occur later. Thus, a new neural-net type algorithm might be used which will predict an optimal push-back schedule integrated over a fifteen minute span. Based upon its output, a controller may delay a given airplane's push-back for 1.5 minutes and another for 2.1 minutes and a third for 0.8 minutes if the algorithm shows that a gain larger than this sum will accrue within the next fifteen minute period.

2. Approach Control Revectoring. In this evaluation scenario the SDTF staff will collaborate in real time with (simulated) TRACON personnel and flight-related input as well as with "local" (i.e., SDTF-derived data) to respond with normally approved procedures to "n" aircraft approaching within a radius of about "D" miles of the airport. During one test period standard (i.e., "traditional") air traffic control procedures will be followed. During another the same staff will use new, candidate, computer-derived rules for dealing with all of the approaching airplanes. For example, a new software program may suggest revectoring thirty percent of these airplanes in order to achieve some desired objective(s) (runway switch; fuel savings; turbulence reduction; conflict avoidance; etc.). The existing tower facilities will be used to display all required information and performance will be carefully recorded and analyzed to identify such things as: error type and rate, inter-personnel communication efficiency and conflicts, problem solving approaches, workload at each controller station, use and misuse of existing equipment, etc.

3. Emergency Approach. This test scenario will involve an inbound airplane with a declared emergency. The same ATC team will take part and the emergency situation will be embedded within an extended number of "N" nominal airplane approaches. Candidate emergency situations might include (landing gear light out; hydraulic system failure; passenger medical emergency; etc.). On each of several repeats of this (single type of emergency) situation the radio call from the pilot will occur at different times "T" before touchdown. For instance, one radio call may occur twenty minutes prior to anticipated touchdown[1], another seven minutes[2], and a third only two minutes. All controller responses will be monitored and recorded for later analysis. A second variable in this two-dimensional experimental design might be coping technique(s): e.g., use of "traditional" method(s) of coping with this kind of an emergency versus "proposed new automated methods for coping ."

4. Approach Sidestep. This test scenario involves a last-minute change of landing runway (in a parallel runway environment). The pilot in command would make the final decision but would be instructed about the existence of an unexpected runway obstruction on the planned runway. Test variables could be amount of time of requested sidestep before touchdown, atmospheric visibility, traffic load, etc.

5. Integration of Automated and Manual Gatekeeper. Since airlines may be future SDTF customers it may be desireable to test SDTF pseudo-ramp controller functions. The Ramp Tower at the Atlanta-Hartfield International Airport (ATL), for example, uses a "Gatekeeper" software program to assist in scheduling airplanes into and out of the many gates. This SDTF validation test would implement the existing ATL Gatekeeper program and vary the incoming and outgoing load so as to require manual scheduling as well. Research variables could include: number of controllers on duty, airplane movement load, and other real-world variables.

Joint NASA/Federal Aviation Administration Program

A recent NASA Headquarters Public Affairs Office release (96-235) was entitled "NASA and FAA Join in Air Traffic Management Research." It cited several specific areas for joint NASA - FAA activities that could impact future usage of the SDTF. They include:

1. Roles of flight crews and ATC. More flexible flight operations may involve new roles for flight crews and air traffic controllers.

2. Air Traffic Management, Cockpit and Fleet Management Integration. Emerging technologies may permit closer integration of air traffic management, cockpit flight management, and operational control centers.

3. Cockpit Situational Awareness. Technologies developed for the Traffic Alert and Collision Avoidance System (TCAS) and Automatic Dependent Surveillance Broadcast (ADS-B) offer new opportunities to improve cockpit situational awareness both on the airport surface and throughout airspace. Further development will increase flight crew participation in air traffic management decision-making.

4. Flight Restrictions. Development of concepts, technologies, responsibilities, and procedures will minimize flight restrictions and maximize aircraft operations. Restrictions will continue to be imposed in high density areas.

5. Safety Analysis. The highest level of safety will be maintained. This will require analysis and simulations of safety hazards and development of tools for proactive detection of potentially hazardous situations.

6. All Vehicle Classes. Flight operations will accommodate all users including but not limited to: transport, general aviation, rotorcraft and military aircraft. The system will accommodate each aircraft class while assuring that avionics requirements are cost effective and affordable.

7. Cost-Benefit Assessments. Each step in the transition to more flexible flight operations will be substantiated by cost-benefit estimates. Projections will include impacts on both airspace users and air traffic management service providers.

Summary

Data obtained in the SDTF may be safely extrapolated to the real world only when validation testing shows that the performance of both the human and automated systems is equivalent to the real environment. This paper presents a variety of validation guidelines as well as general and specific testing approaches which can be used to generate these needed data.

References

Bikson, T.K., B.A. Gutek, and D.A. Mankin, *Implementing computerized procedures in office settings: Influences and outcomes.* The RAND Corp., Report R-3077-NSF, Santa Monica, Calif. 1987.

Boff, K.R., and J.E. Lincoln, *Engineering Data Compendium: Human Perception and Performance.* Harry G. Armstrong Aero-space Medical Research Laboratory, Wright-Patterson Air Force Base, Ohio, 1988.

Haines, R.F., A system performance throughput model applicable to advanced manned telescience systems. Research Institute for Advanced Computer Science, NASA-Ames Research Center, *Technical Report 90.1*, January 1, 1990 (a).

Haines, R.F., Human performance measurement-validation procedures applicable to advanced manned telescience systems. Research Institute for Advanced Computer Science, NASA-Ames Research Center, *Technical Report 90.10*, February 12, 1990 (b).

Harwood, K., Defining human-centered system issues for verifying and validating air traffic control systems. In Wise, J., V.D. Hopkins and P. Stager (eds.), *Verification and Validation of Complex and Integrated Human Machine Systems.* Springer-Verlag, Berlin, 1993.

Helander, M., (Ed.), *Handbook of Human-Computer Interaction.* North-Holland, New York, 1988.

McGuigan, F.J., *Experimental Psychology: A Methodological Approach.* Second Edition, Prentice-Hall, Inc., Englewood Cliffs, New Jersey,1968.

Tillman, P., and B. Tillman, *Human Factors Essentials: An Ergonomics Guide for Designers, Engineers, Scientists, and Managers.* McGraw-Hill, Inc., New York, 1991.

Wise, J., V.D. Hopkins, and P. Stager (eds.), *Verification and Validation of Complex and Integrated Human Machine Systems.* Springer-Verlag, Berlin, 1993.

Woodson, W.E., B. Tillman, and P. Tillman, *Human Factors Design Handbook*, Second Edition, McGraw-Hill, Inc., New York, 1992.

Footnotes

1. This radio communication would (likely) come to the tower via airline operations, ARTCC, or TRACON.
2. This radio communication would come to the tower via airline operations, TRACON, or city operations.

AIAA-97-3668

TESTING SAR MOTION MEASUREMENT FLIGHT CODE IN A UNIX ENVIRONMENT

Theodore J. Kim*

Aided Navigation and Remote Sensing Department
Sandia National Laboratories
Albuquerque, NM 87185-0843

Abstract

Measuring the motion of a synthetic aperture radar antenna requires inertial navigation algorithms, a Kalman filter to update the navigation solution using GPS measurements, and extrapolation to minimize the latency of the motion solution provided to the radar. A software-only simulation environment is described that complements hardware-in-the-loop testing and allows engineers and analysts to test the flight code in a setting where data storage and analysis are straightforward. The flight code and simulation, consisting of a 6-DOF aircraft model, inertial measurement unit model, and GPS satellite/receiver model, run many times faster than real-time on a desktop UNIX workstation.

Introduction

Synthetic Aperture Radar (SAR) image formation algorithms require very accurate knowledge of the antenna motion in order to produce an image that meets specifications. An unmeasured vibration with an amplitude of less than a millimeter can produces sinusoidal position errors that raise impulse response sidelobes to unacceptable levels.

Sandia's approach to motion measurement (MoMeas) is to place an inertial measurement unit (IMU) as close as possible to the SAR antenna. Since most SAR antennas are gimballed to allow pointing at various targets, the IMU is usually mounted on the inner-most gimbal along with the antenna. Figure 1 shows the Department of Energy's DeHavilland Twin Otter research aircraft and the location of the antenna radome. The 3-axis gimbal, SAR antenna, and IMU that are currently being used on the airplane are shown in Figure 2.



Figure 1: The Department of Energy's DeHavilland Twin Otter.

The SAR MoMeas software must be tested before flight to reduce flight test costs and program risk. Hardware-in-the-loop (HIL) testing consists of running the MoMeas software on the actual flight computers while accepting simulated or real data from flight tests over the actual interfaces.

While HIL testing must be performed, (1) it is not an environment that is conducive for algorithm development, (2) data storage can be difficult, and (3) limited hardware resources often limit the time available to test the MoMeas software. Therefore, an environment is desirable that is easy to use and doesn't require the flight hardware. Additionally, the environment should accept the majority of the flight code intact so that multiple versions of the code do not exist for different testing setups. An environment called software-only simulation, or SOS, that runs on a desktop UNIX workstation satisfies the above criteria.

Before the SOS environment is discussed, the SAR MoMeas algorithms will be reviewed and the simulations that are used to generate the inputs to the MoMeas code will be explained. The setup of the real-time code is then presented, followed by a description of the SOS environment. Finally, the overall benefits of the SOS environment are summarized.

*Senior Member of the Technical Staff, Senior Member AIAA.

Copyright ©1997 by Theodore J. Kim. Published by the American Institute of Aeronautics and Astronautics, Inc. with permission.

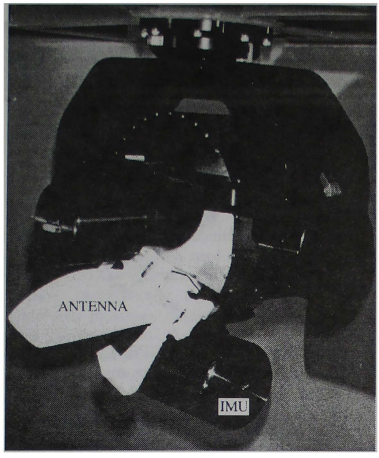


Figure 2: Three-axis gimbal with antenna and IMU.

SAR MoMeas Algorithms

Mounting the IMU and the SAR antenna at the same location, as shown in Figure 2, produces a reliable, straight-forward, and arguably the most accurate implementation for measuring the motion of a SAR antenna. The delta-velocity and delta-angle integer counts coming from the IMU are used to generate a navigation solution and stabilize the antenna[1]. Various measurements from a GPS receiver are used to update the navigation state via a Kalman filter[2] (see Figure 3). This updated navigation solution is used to help the aircraft pilot stay within 50 m of a nominal flight path.

Before the motion solution is provided to the radar, it is extrapolated forward in time using maximum entropy extrapolation[3, 4] to minimize the latency of the data. This extrapolation process uses the previous 75 range measurements that have been calculated from the GPS-aided navigation solution. While most GPS-aided navigation solutions have discontinuities that arise when the updates are applied, Sandia's MoMeas system uses a unique approach that avoids discontinuities without filtering the navigation solution, which would introduce unwanted latency[1].

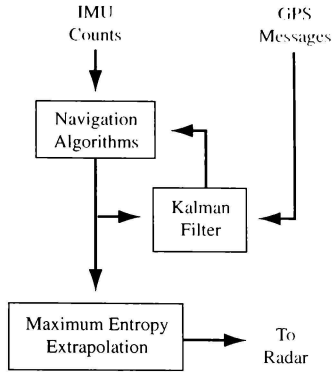


Figure 3: MoMeas algorithms.

Simulations

Both HIL and SOS environments require IMU counts and GPS measurements as inputs to simulate the hardware outputs during different flight conditions. A simulation consisting of a six degree-of-freedom (6-DOF) aircraft model, an IMU model, and a GPS model is used to generate the necessary inputs.

The 6-DOF simulation uses three guidance loops to produce data that correspond to different flight profiles. The different loops control the heading, velocity, and altitude of the aircraft. The heading loop implements bank-to-turn guidance – a roll moment is commanded based on the heading error. Limits on the maximum bank angle, roll rate, and roll acceleration are user-selectable to emulate various aircraft. Velocity matching is performed by changing the x-body force and altitude is maintained by commanding a vertical force. Pitch and yaw moments are commanded to keep the body pointed along the velocity vector.

The guidance commands are scripted based on time and are read from a data file at runtime. The three guidance loops allow simulation of an entire flight profile which may include ground alignment, takeoff, climb to altitude, maneuvers on station, and in-air alignment.

The 6-DOF simulation produces the state of the aircraft (position, velocity, and attitude) as well as the specific forces and angular velocities sensed by the plane. These specific forces and angular velocities from the 6-DOF simulation are transformed to

the IMU frame via a three-axis gimbal model. Various IMU instrument errors, quantization errors, and frequency response characteristics are simulated to produce the IMU counts. These counts are used as inputs to the SAR MoMeas flight code.

The GPS measurements are simulated using the true state of the aircraft from the 6-DOF model (since the GPS antenna is mounted to the aircraft body) and the almanacs for the GPS satellites, which are updated weekly and can be downloaded from the Coast Guard web site[†]. Measurement noise can be simulated or derived from static or dynamic data. A satellite selection algorithm emulates the algorithm implemented in the GPS receiver and antenna masking information accounts for obstructions in the field of view and the attitude of the aircraft.

The simulations are verified by examining 2-dimensional plots of the outputs and 3-dimensional graphical visualizations of the data. Screen shots of the aircraft/SAR antenna and GPS satellites/antenna masking are shown in Figures 4 and 5. The codes were written using the OpenGL language and are simple enough to be maintained and modified by the engineers involved with the development of the simulations and flight code.

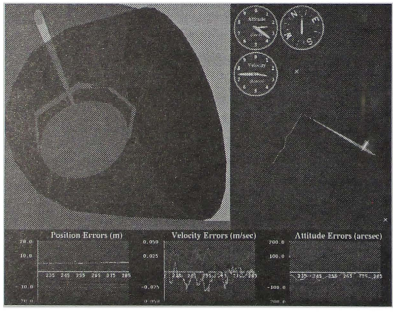


Figure 4: Screen capture of aircraft visualization showing the SAR antenna, groundtrack, and navigator errors.

Software Setup

The MoMeas software is written using ANSI-standard C and runs on multiple processors of the flight computer. The combination of the operating system and the types of processors that are used on some commercial-off-the-shelf (COTS) systems

[†]http://www.navcen.uscg.mil/gps/gps.htm

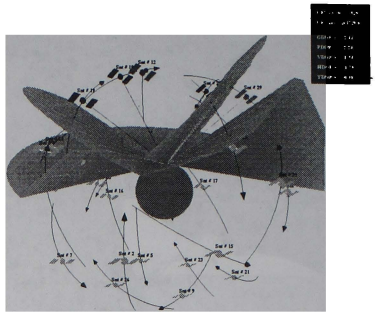


Figure 5: Screen capture of GPS visualization showing GPS satellites and masking angles for two GPS antennas.

can cause the individual processors to have poor interrupt performance. For this reason, the MoMeas code has been structured using a "polling" architecture. Typically, a 680x0 processor handles input and output interfaces and notifies the waiting processors when the data is available. While this software architecture is sub-optimal for the real-time code, it does lend itself well to the SOS environment.

The setup of the code for SOS is shown in Fig. 6. The majority of the SAR MoMeas flight code is used without changes. This code has been surrounded by the "wrapper" code that is necessary to get the code to function in a UNIX environment. The wrapper code replaces the "main" and initialization functions that are associated with each of the flight computer processors. Less than 10 lines of code have been added to the rest of the flight code to get the code to function under UNIX. A flag set in the makefile activates these sections of code when it is compiled as an executable.

The simulations are also written in C and compiled as a separate executable. Compiling the MoMeas code separately from the simulation eliminates possible conflicts between global variables and keeps the codes more organized. The MoMeas and simulation codes execute sequentially in a loop on an 192 MHz R10000 Silicon Graphics (SGI) Indigo^2 workstation. SGI-specific Inter-Process Communication (IPC) is used to transfer data between the two executables since the vendor claims it is faster than UNIX IPC, which is platform-independent.

Although the MoMeas code can be compiled as separate executables to test the multiple processes

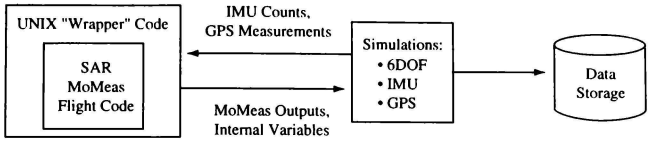


Figure 6: Software-Only Simulation data transfer.

running in the real implementation, the cost of running multiple MoMeas executables on a single-processor UNIX workstation is much slower execution time. Additionally, the interactions between the MoMeas processes on the UNIX machine will not necessarily emulate the real-time hardware and those interactions will be tested in the HIL environment and during flight test. Currently, the SOS environment runs approximately seven times real-time on the aforementioned computer with about 15% of that time used to switch between the simulation and MoMeas executables.

Real IMU and GPS data that has been collected during flight tests can also be used instead of the simulation. The MoMeas code has been run using this data to reproduce problems encountered during flight test and re-tune the Kalman filter. The only disadvantage when using the real data is that no "truth" data exists since most scoring systems do not produce solutions that have the millimeter-level accuracy that is necessary to analyze SAR performance. When the flight data is used instead of the simulation, the SOS environment runs over twelve times real-time.

Conclusions

Software-only simulation (SOS) is a UNIX environment that is used to test SAR Motion Measurement (MoMeas) flight code. The inputs that are necessary for the MoMeas code come from either a high fidelity simulation that includes an aircraft model, an inertial measurement unit (IMU) model, and GPS satellites/receiver model or data that was recorded during flight tests. The SOS environment complements hardware-in-the-loop (HIL) testing since the algorithm design, analysis, and testing can be performed using the numerous tools that are available for UNIX machines while the HIL testing can focus on the hardware implementation. The relative simplicity of the SOS environment minimizes the number of people necessary to maintain it and allows the analysts to change the actual flight code.

Acknowledgment

Many people have been involved with the development of Sandia's MoMeas algorithms. Some of the people who have greatly impacted this area are Scott Burgett, Rick Fellerhoff, Jeff Gilkey, Arnel Oczon, Todd Owen, Mike Pedroncelli, Ron Tucker, and Al Watts. Jeff Gilkey was the first at Sandia to develop 3-dimensional visualization code to analyze navigation performance.

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy under Contract DE-AC04-94AL85000.

References

[1]Fellerhoff, J. R., and Kohler, S. M., "Development of a GPS-Aided Motion Measurement, Pointing, and Stabilization System for a Synthetic Aperture Radar", Proceedings of the 48th ION Annual Meeting, Washington, D. C., June 29, 1992.

[2]Owen, T. E., et. al., "High Accuracy Integrated Global Positioning System/Inertial Navigation System LDRD: Final Report", Sandia Report, SAND97-0674, March, 1997.

[3]Makhoul, J., "Linear Prediction: A Tutorial Review", Proceedings of the IEEE, Vol. 63. No. 4, April 1975.

[4]Press, W. H., et. al., **Numerical Recipes in C**. Cambridge University Press, New York, NY, 1992.

SPACE-BASED, ULTRA LIGHTWEIGHT IMAGING TECHNOLOGY EXPERIMENT MODELING IN THE SPACECRAFT SIMULATION TOOLKIT

Dr. Richard K. de Jonckheere
Air Force Phillips Laboratory, Space System Technologies Division/VTS
Kirtland AFB, NM
Benjamin L. McGlamery
Daniel Mergens
Terri Franklin
Photon Research Associates, Inc.
San Diego, CA

Abstract

The Spacecraft Simulation Toolkit (SST) architecture is an advanced, flexible, software development framework for modeling and simulation (M&S) of spacecraft based upon state-of-the-art modeling techniques, a visual programming and simulation environment, and accurate physical phenomenology. The SST is providing simulation support to an internal Phillips Laboratory program, the Ultra Lightweight Imaging Technology Experiment (UltraLITE). The UltraLITE program is researching and demonstrating enabling technologies associated with the control of space-based, large aperture, sparse optical arrays. The focus of the program is on precision wavefront control of sparse optical arrays by demonstrating methods for sensing wavefront aberrations, converting sensed wavefront aberrations into telescope control requirements, and controlling the mirror elements with nanometer range actuation. A sparse optical array M&S capability, receiving inputs from the proposed UltraLITE experiments, will provide performance data of the UltraLITE control architecture applied to the different sparse array configurations. This M&S capability will also relate payload performance data to military utility of proposed systems. The initial effort, covered in this paper, includes the development of a prototype "proof of concept" UltraLITE simulation for the single secondary Golay-6 within the SST architecture.

Introduction

The SST is being developed under a Small Business Innovative Research (SBIR) program with Photon Research Associates and managed by the Air Force Phillips Laboratory (PL). The mission of PL's Space Technology Directorate is to provide innovative space technology solutions applied to satellite surveillance, communication, and navigation systems. Computer-based M&S of such complex space systems is critical to reducing the cost, risk, and time to build them. This approach will support the systems acquisition phase, space operations, satellite control, and training of the satellite operators. PL is using the requirement of reducing the cost of space systems development, acquisition, and operations to drive the development of advanced M&S capabilities, overlaying these with Department of Defense (DOD) promulgated standards and specifications, and delivering space M&S products to the customer.

Integrated, modular, and easy-to-use digital M&S of the end-to-end performance of spacecraft sensors, structures, and supporting subsystems can provide powerful insights into the design, integration, and operation of satellites and related spacecraft weapon and remote sensing systems. Such simulations can support a wide range of functional capabilities in the design, development and management of the operational life cycle of such spacecraft. The rapidly evolving capabilities of computer workstations, new system programming environments, and advanced graphics and user interfaces provide the capacity to integrate separate computer models operating in different environments into high-fidelity interactive system simulations.

SST Architecture

The SST is an advanced, flexible, development environment for the modeling of spacecraft, the environment in which it resides, and the dynamic effects of environmental interactions on the spacecraft. The SST is based upon state-of-the-art simulation methods and accurate physical phenomenology. It is an object-oriented system[1] consisting of software objects which simulate the various systems and subsystems of the physical spacecraft.

This paper is declared a work of the U.S. Government and is not subject to copyright protection in the United States.

Khoros, developed by Khoral Research Inc. (KRI), is the simulation software selected as the basic working environment for the development of the SST. Through Khoros, the SST provides the ability to visually and interactively integrate the software objects together into a simulation of either a complete spacecraft system or a spacecraft subsystem. This visual environment allows the SST user to access spacecraft system, payload, or subsystem models through pull-down menus, and then connect the modules using a mouse into the configuration required to implement a simulation. Figure 1 shows the SST visual environment. Figure 2 shows the basis of the SST architecture design. The interactive environment provides a number of other supporting functions such as integrated data analysis and software development tools. The user or model developer virtually builds a spacecraft by selecting and integrating components, duplicating the equivalent real world actions.

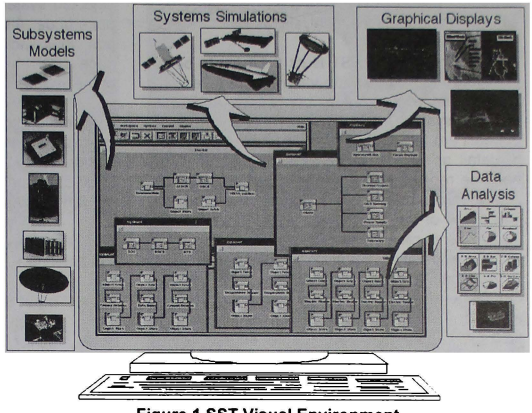


Figure 1 SST Visual Environment

The SST software objects provide algorithmic simulation of the various spacecraft functions. The simulation data bases provide the necessary knowledge base within which the detailed characteristics of the system are described, and an or-

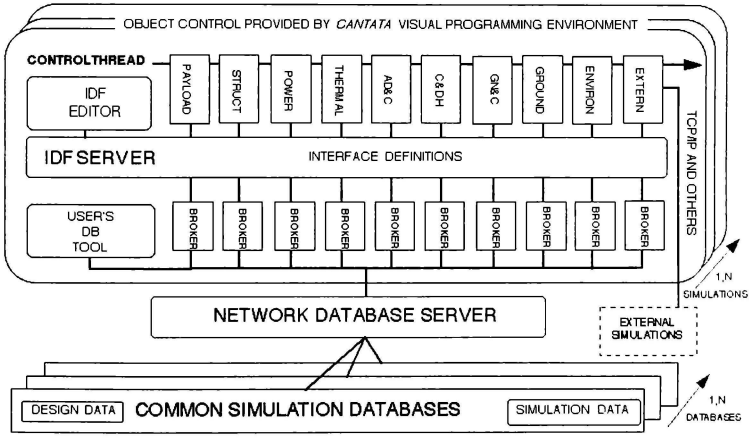


Figure 2 SST Architecture

derly and efficient means to store the results of a simulation for additional analysis.

A key feature of the SST is its flexibility to be reconfigured to meet a wide variety of requirements in engineering, simulation, operations, and training. Subsystem representations of varying levels of detail are, or can be made available. This level of detail as well as whether a given subsystem need be represented at all for a given application can be tailored depending upon the intended use. The ability of the SST to be reconfigured to meet requirements ranging from detailed engineering to real-time operations/training means that the system can support the broadest range of simulation requirements within a single, integrated environment. The characteristics of the simulation can also be varied, even during the course of the simulation, in order to balance the desired simulation goals with the available computing resources.

When used to support the typical development of a spacecraft system, the SST user will initially configure the system with generic representations of various subsystems chosen from the SST software library. These can be used to explore design options and to produce initial proof-of-concept studies. As the design process proceeds, the virtual spacecraft being built in the SST will evolve as well. Certain specialized subsystems may not be represented in sufficient detail through existing software, so new software may be custom written and integrated. Through the use of custom software and highly customizable parameter inputs for generic subsystems (such as thermal and structural design software) the SST will increasingly represent a precise working version of the ultimate spacecraft system.

Implementation Example

We present here an example of the above concepts. In this example the objective is to take a specific design of a sparse aperture optical system and simulate the images it would produce over a range of operational conditions. Because the optical system produces blurred images, an important part of the simulation process is to include a deconvolution process to enhance the images. The resulting images can then be assessed via qualitative and quantitative methods to determine if the selected system will provide the desired performance under the test conditions. Quantitative methods would include a suitable measure of performance (MOP) based on measurements and analysis of the images.

As mentioned previously, the system we have chosen to model is one of several under consideration within the UltraLITE program. The detailed technical aspects of this program are provided in the References[2,3,4].

The necessary elements of the system which are simulated are:

- Input Scene
- Mirror Configuration
- Mirror Displacements
- Point Spread Function (PSF)
- Convolution of the PSF and Scene Image
- Sensor Effects
- Deconvolution
- Measures of Performance

The significance of each of these elements and their simulation will be described, and representative examples will be shown.

Input Scene

The input scene, in digital form, should be representative of the type of scene to be imaged by the optical system, in terms of wavelength regime and scene content. Ideally it is a calibrated scene in absolute radiance units. The resolution of the scene needs to be higher in resolution than the optical system and sensor to avoid aliasing effects.

Calibrated input scenes can be obtained either from calibrated camera scenes or by camera scenes which have been reduced to images that have been classified according to materials types. The material images, along with terrain elevation, can then be used to render images from any viewpoint and for any optical wavelength. Figure 3 shows a scene created in this manner. It is an image of a hanger area at Edwards Air Force Base, California. The scene has been rendered at a wavelength of .55 micrometers. The radiance of the atmosphere for an overhead nadir viewing sensor is included in the scene.

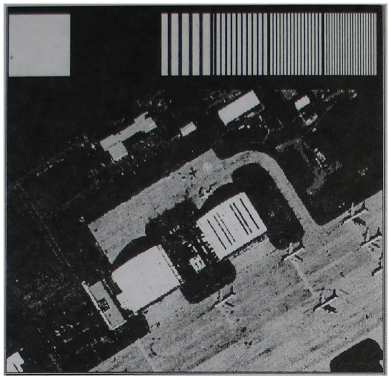


Figure 3 Radiometrically Calibrated Scene

The scene also contains a test pattern inset for computing MOPs. This test pattern was made with standard Khoros functions.

Mirror Configuration

For a high quality optical system the resolution is the order of λ/D, where λ is the center wavelength of the system passband and D is the diameter of the primary mirror. The diameter required for a conventional system can lead to impractical weight and rigidity requirements for space observatories.

A solution to this problem is to use sparse aperture systems which have the potential for nearly the same resolution as a conventional filled aperture at a fraction of the weight of the conventional system. It achieves this by providing a sampling of the complete range of the spatial frequencies of the monolithic mirror but at the expense of reduced collected flux, and significantly attenuated spatial frequencies due to the sparseness of the aperture.

Another aspect of the light-weighting of a sparse aperture system is that the individual mirrors are not mounted on a rigid structure. Rather active mirror control is used to keep the mirrors on an ideal surface as structural elements in the observatory are displaced by vibration and maneuvers. There are a number of sparse aperture mirror configurations. For this study we have chosen the Golay 6 system[5], as shown in Figure 4.

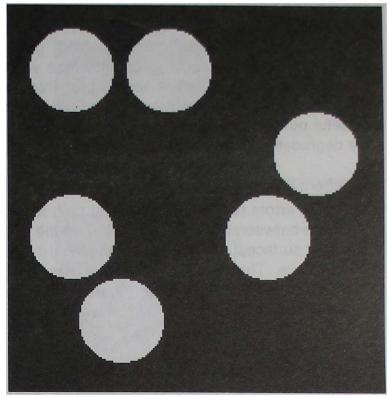


Figure 4 Golay 6 Mirror Configuration

Sparse Aperture PSF and MTF

The point spread function (PSF) associated with the sparse aperture system is needed for system analysis. For design purposes an optics code would typically be used to compute the PSF as a function of location in the field-of-view. However, for the purposes of concept exploration, a faster but less general method is sufficient. To use this method the following assumptions are made: 1) The image sensor responds to a narrow bandwidth of light (near-monochromatic assumption), 2) The paraxial assumptions frequently used in Gaussian optics analysis are satisfied, i.e. $\sin(\theta) \approx \tan(\theta) \approx \theta$, where θ is the angle of the most extreme ray with respect to the optical axis.

If these assumptions are satisfied, then it can be shown that, apart from a quadratic phase error, there is a Fourier transform relationship between the complex wavefront at the exit pupil of a lens and the image in the focal plane of the optical system[6]. If the object is a point source at infinity, the image is the coherent PSF.

Assuming the scenes of interest for the observatory are illuminated by incoherent light, the incoherent PSF is relevant. This is found by computing the modulus square of the coherent PSF.

The Fourier transform of the incoherent PSF produces the Optical Transfer Function (OTF), which describes the amplitude weighting and phase shifts of the spatial spectrum of an image when convolved with the PSF. The Modulation Transfer Function (MTF) is the modulus of the OTF and is a useful quantity for assessing the degree of image degradation in the frequency domain.

Wavefront Errors

Wavefront errors represent the optical path length difference between the light reflected from the mirror array surfaces and the ideal spherical reference surface. The errors have two origins: fixed errors due to the mirror surface (figure errors and aberrations), and mechanical displacements of the mirrors.

The mirror displacements consist of x, y, z translations and rotations about the x, y, and z axes, where z is perpendicular to the optical axis of the telescope. Typically the most significant displacements are in the z direction, commonly called piston motion, and rotations about the x any y axes, commonly called x and y tilt. (In low F number systems the x and y translations and rotation about z can be significant .)

The path length differences are multiplied by 2 to account for the two way path for a reflective optical element. The path length differences are next converted into wavefront phase errors, for a single wavelength of light, by multiplying by $2\pi/\lambda$. A complex wavefront, covering the extent of the monolithic mirror is computed from this data.

The SST procedures for implementing the piston and x and y tilts for a single mirror are shown in Figure 5. The piston and tilt errors are supplied to this procedure from displacement files.

The collection of procedures shown in Figure 5 is replicated for each mirror to produce the complex wavefront for the Golay 6 array, as shown in Figure 6. A Fourier transform and modulus square operation complete the operations needed to compute the coherent PSF. Figure 7 shows a representation of the path length errors induced on the wavefront by piston and tilt errors.

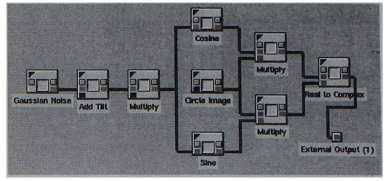


Figure 5 Single Mirror Procedure

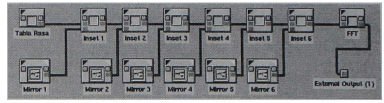


Figure 6 Coherent PSF Procedure

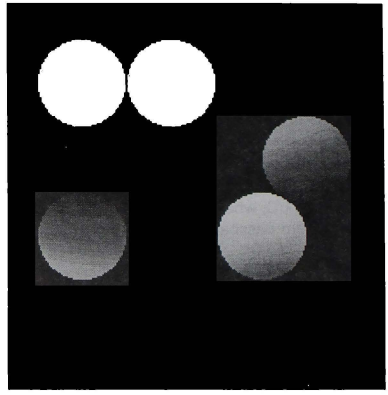


Figure 7 Piston and Tilt Mirror Displacements

Figure 8 compares PSF's and MTF's for three cases: 1) a monolithic mirror which would enclose the Golay 6 array, 2) the Golay 6 array without mirror displacements, and 3) the Golay 6 array with mirror displacements of .15 waves RMS error. The significant attenuation of the MTF at high spatial frequencies for the Golay 6 mirror in comparison to the monolithic mirror is apparent. However, the advantage of the phased array is evident in that the amplitude of the MTF, after the initial drop, is nearly constant out to the diffraction cutoff frequency of the enclosing monolithic mirror.

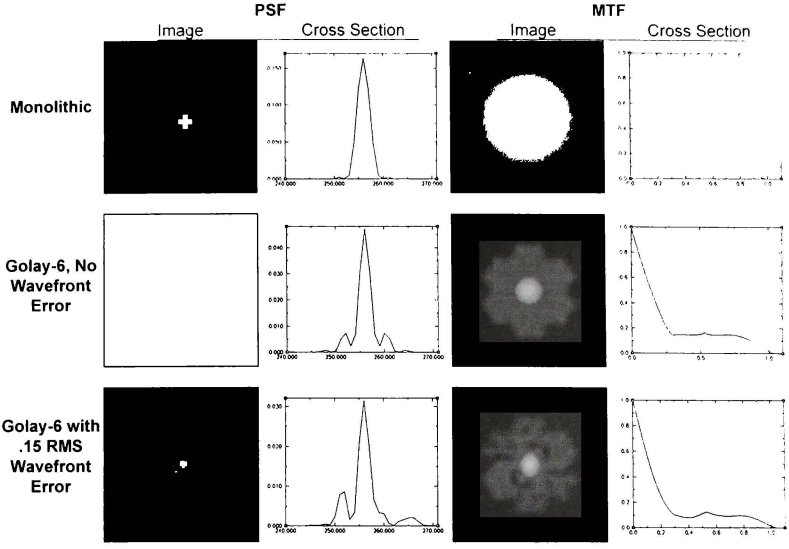


Figure 8 Monolithic and Sparse Aperture PSF's and MTF's

Convolution With Scene

Given the assumptions made previously, the PSF is shift invariant, i.e., it has the same shape in all locations in the field of view. The image formed by the optical system of the object-space scene can be computed by a convolution of the scene and the PSF. Since the PSF is shift invariant, the convolution can be performed by Fourier transform methods.

Sensor Effects

At this point the calculation has produced the focal plane image. The next step in the simulation process is to add effects of the sensor. The primary effects are an additional blurring and the addition of noise. The nature of these effects will vary depending on whether the sensor is a staring focal plane array or a scanning array, possibly with time delay integration, and on the intrinsic detector material and design.

Detailed models for the simulation of both staring and scanning[7] sensors have also been designed for integration within the SST environment.

Deconvolution

As demonstrated in Figure 8, a characteristic of a sparse aperture array is that the modulation transfer function over a considerable range from the low to high frequencies is significantly attenuated in comparison to the MTF of a monolithic mirror. To attain resolution performance approaching the monolithic mirror, the Golay 6 MTF must be amplified by filtering the sensor output image to boost these attenuated frequencies. However, noise becomes a significant factor with increasing spatial frequency, and must be dealt with. A variety of image processing filters can be used to perform this function. We have chosen the Wiener filter as an example[8]. The basic process is to Fourier transform the sensor output image, multiply by the Wiener filter, then inverse Fourier transform to obtain the deconvolved image.

The Wiener filter is based on the mathematical approach of minimizing the mean squared difference between the undegraded noise-free image and the Wiener filtered image. The Wiener filter is basically an inverse filter multiplied by a factor whose value is essentially unity when the ratio of the noise Power Spectral Density (PSD) to the sensor image PSD is small, but becomes small when the ratio becomes large, thus preventing large amplification in those regions of the spectrum of low signal to noise ratio.

The Wiener filter requires three inputs: the Fourier transform of the PSF, the noise PSD, and the noise-free sensor output image PSD. Approximations may be needed for each of these.

The Wiener filter has been applied to a Golay 6 blurred image as an example of its ability to reduce the effects of blur and noise. Conditions simulated in this example are:

Wavelength = .55 micrometers
Individual mirror diameter = 2.2 meter
Effective monolithic diameter = 9 meter
Altitude = 35768 kilometers
Altitude*λ/Deff = 2.2 meter
Scene pixel resolution = .76 meters
Wavefront error = .15 waves
Noise Equivalent Reflectance = 1%

The spectrally white noise added to the scene was a Noise Equivalent Radiance equal to the radiance of a unit lambertian reflectance X Noise Equivalent Reflectance (%) X 100.

For the Wiener filter, the noise PSD was computed from the knowledge of the RMS value of the additive white noise. The PSD of the input image was represented by a radial average of the actual image PSD, thus removing any detailed *a priori* knowledge of the two dimensional image power spectrum.

The results are shown in Figures 9 -11: 9) is the blurred image before deconvolution, 10) is the filtered image using the exact PSF, i.e. the PSF with both sparse aperture and wavefront error effects included, and 11) is the Wiener filtered image using only the known Golay 6 PSF.

Measures of Performance (MOP)

MOPs are needed to determine if system performance is adequate. Images produced by the above simulation and processing steps can be assessed visually to give a subjective MOP. In addition, quantitative measurements can be applied to test patterns which are subjected to the same processing as the images. The patterns can either be embedded in the scene image or can be processed separately if the processing is done with the same parameters as used for the image. Noise

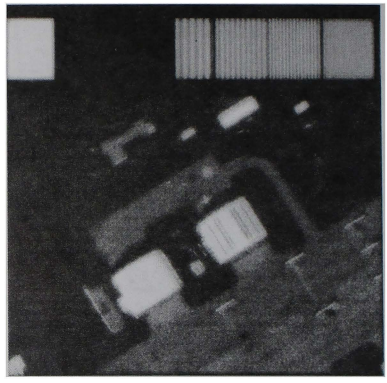


Figure 9 Blurred Image.15 RMS WFE, 1% Noise

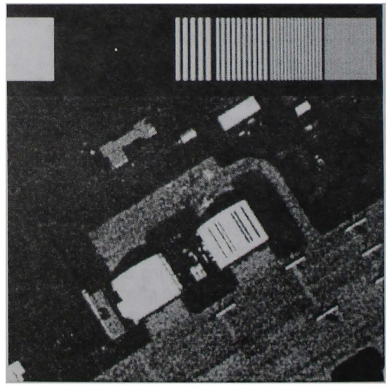


Figure 10 Wiener Filtered Image Using Blurring PSF

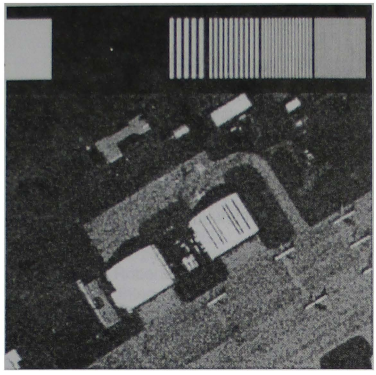


Figure 11 Wiener Filtered Image Using Only Golay 6, PSF

can be omitted from the test pattern or can be added, depending on the objective of the test.

As shown in previous figures, the example scene used has an embedded test pattern. The features of this test pattern are an edge, an isolated single bright pixel, and four sinusoidal patterns. In the blurred image the edge allows measurement of edge response functions. The single pixel, when blurred, becomes a PSF. From the PSF a number of MOPs can be computed: Strehl ratio, PSF width, and, via a Fourier Transform, spatial frequency parameters such as MTF and restorable bandwidth.

Figure 12 shows examples of MOPs from a trade study of wavefront errors of .10,.15, and 20 rms waves and Noise Equivalent Reflectance's of .5%, 1% and 2%. The MOPs used were:

1. Edge gradient - 20% to 80% amplitude width of edge.,
2. PSF Width - 50% amplitude width, equivalent to Rayleigh criteria.,
3. Restorable bandwidth (fraction of frequency at which Wiener filter attenuator component = .5)

Data for these MOPs were extracted from the simulated data using standard Khoros routines. The plot tool, xmgr, was used for data analysis and presentation.

The results are examples of how the SST can be used to perform a wide variety of functions for imaging system performance analysis with minimal programming effort.

Summary

In this work we have demonstrated that with the appropriate environment, technically complex satellite imaging concepts can be rapidly prototyped and tested.

The simulation environment is the Spacecraft Simulation Toolkit, developed for the virtual prototyping of spacecraft and spacecraft interactions with the natural environment.

The example presented in this paper is the simulation of a Golay 6 sparse aperture system. Nearly all major effects of the imaging, process, sensor noise, and deconvolution were performed with minimal programming effort. It was demonstrated that extraction of data for the computation of MOPs can be accomplished in the same environment.

Acknowledgments

The authors would like to thank the Phillips Laboratory UltraLITE development team for introducing us to the UltraLITE concept and providing many valuable suggestions regarding simulation of their system concepts documented in the references[1,2,3].

We also thank Leif Hendricks, Manager of PRA's Innovative Infrared Scene Generation Project, Sally Westmoreland, Analyst, and Keem Thiem, Edwards Air Force Base Program Manager, whose program provided us the scene, classified according to materials types, of Edwards Air Force Base, from which we obtained the visible band radiometrically calibrated test scene.

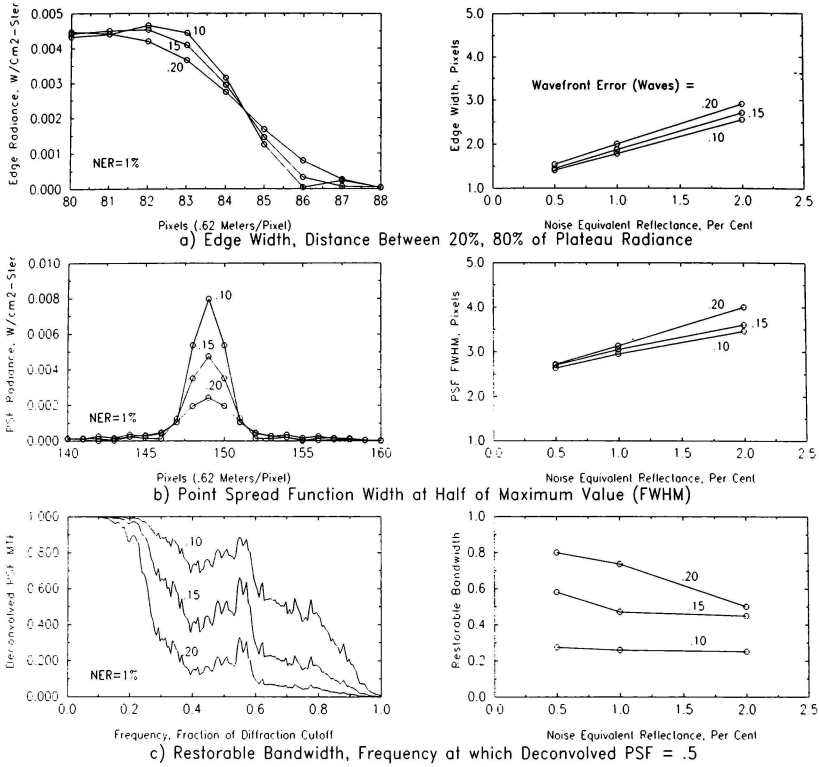


Figure 12 Examples of Measures of Performance from Trade Study. Left plots are from Wiener filtered images, shown only for one noise level. Right plots are MOP data extracted from left plots for all three noise levels.

References

1. J. Rumbaugh, et al, "Object-Oriented Modeling and Design", (Prentice Hall, 1991).
2. Mike Powers, Jesse Leitner, Kevin D. Bell, Rich Boucher, Lawrence Robertson, and Karl Schrader, "Assessment of a Large Aperture Telescope Trade Space and Active Opto-Mechanical Architecture", Phillips Laboratory/Space Experiments Directorate, 1997, Mike Powers, powersm@plk.af.mil .
3. Lawrence Robertson, Jesse Leitner, and Brett de Blonk, "Integrated Modeling and Control of the UltraLITE System" Phillips Laboratory/Space Experiments Directorate, 1997, Mike Powers, powersm@plk.af.mil .
4. Kevin D. Bell, Richard H. Boucher, Robert Vacek, Michael Hopkins, "Assessment of Large Aperture Lightweight Imaging Concepts", Phillips Laboratory/Space Experiments Directorate, 1997, Mike Powers, powersm@plk.af.mil .
5. Marcel J. E. Golay, "Point Arrays Having Compact, Nonredundant Autocorrelations, J. Opt. Soc. Am. , 61, 272 (1971).
6. Joseph W. Goodman, "Introduction to Fourier Optics", (McGraw-Hill, 1968), pp 83-89.
7. Jeffrey Johnson, "Scanning CCD with TDI Sensor Model", Photon Research Associates, redhead@photon.com , 1997.
8. C.W. Helstrom, "Image Restoration by the Method of Least Squares", J. Opt. Soc. Am. 57. 297, (1967).
9. Richard de Jonckheere, Terri Franklin, David Wilson, "Spacecraft Simulation Toolkit: An Advanced Distributed Simulation Framework for Supporting the Space Warfighter". Phillips Laboratory, 1997.

AIAA-97-3670

CONTINUOUS LINEAR IMPULSE BALANCE METHOD FOR REAL-TIME SIMULATIONS OF CONTACT DYNAMICS

F.J. Belanger* and X. Cyril†

CAE Electronics Ltd.
C.P. 1800, St-Laurent, Canada, H4L 4X4

The continuous impulse balance method provides for efficiently modeling impact and continuous contact by combining features from the time continuous and impulse balance procedures. It is introduced in general form by considering the two-body contact dynamics problem. External forces are included in the formulation when invoking the impulse balance principle to calculate the contact forces. This allows for more accurately handling impact and, more importantly, makes it possible to model continuous contact in a completely seamless fashion. Global momentum conservation is enforced during the collision, subject to a form of mechanical energy conservation involving the coefficient of restitution. The method does not require that the time step be refined during the collision by prescribing a profile for the contact force that recognizes the need to offset the inaccuracies associated with non-conservative effects. Thus, the time step stays constant throughout the simulation. Furthermore, the method is not iterative and maintains linearity in slip-contact situations by appropriately defining the slip direction.

Introduction

THE robustness of candidate contact dynamics methods can be assessed by their ability to cope with both *impact* and *continuous contact*, where the latter involves two bodies that push and slide past each other without relative velocity of approach or separation. Two procedures that address with relative merit these two cases are commonly used in numerical studies: (1) the continuous time method, and (2) the impulse balance method. (In this paper the term "contact" generically encompasses both impact and continuous contact.)

The continuous time method can model both impact and continuous contact. The contact forces are assumed to act in a continuous manner, and this allows for conducting the analysis of a system simply by adding them to the equations of motion. Widely used approaches for calculating the contact forces make use of Hertz contact law, for example, to relate them to the material deformations of the surfaces.[1] Contact deformations, however, are dependent on the geometrical features of the surfaces, and these often can only be approximately determined. The accuracy of solutions can be improved—at the expense of considerable computational overhead—by resorting to full finite element analysis of the colliding bodies.[2]

In the continuous time method it is necessary to refine the time step during the collision to capture the small time scale of contact dynamics. This restricts the ease with which the different approaches can be implemented in real-time simulations, and necessarily places additional demands on the computational resources. Furthermore, the problems are compounded by the approaches often being iterative and non-linear, especially during continuous contact.[3]

The continuous time method enforces momentum conservation on the system because the contact forces, whether accurately calculated or not, are applied on both bodies in action-reaction fashion. Increased accuracy of the contact forces results in better predictions for the post-impact configuration of the system (relative position, velocity, orientation, etc., of the bodies), but does not affect global momentum conservation.

The degree of elasticity of the collision is another factor which affects post-impact system configuration. It involves the additional consideration of conservation of mechanical energy, and almost invariably necessitates for practical modeling purposes that empirical data be introduced. The coefficient of restitution is the most widely used such data. It allows for lumping the overall effect of non-conservative forces into a single coefficient which is a measure of the relative importance of the contact forces during the deformation and restoration periods.[4] Its use proved effective when combined with Hertz contact law to model non-linear effects such as hysterisis.[1] Finite element methods must deal with non-conservative forces in a more complicated way, for example by introducing modal damping coefficients.[2] These, however, are just as hard if not harder to estimate than the single coefficient of restitution, and they are more difficult to relate to the physics of the system.

Copyright © 1997 by CAE Electronics Ltd. Published by the American Institute of Aeronautics and Astronautics, Inc. with permission.

*Project Engineer, Space Systems Engineering.

†Manager, Space Systems Engineering.

The impulse balance method, on the other hand, can only model the impact phase of contact dynamics, and therefore has a more limited scope of applicability than the continuous time method. It is not necessary to calculate the contact forces. Rather, the method directly determines the post-impact velocities of the bodies by enforcing global momentum conservation; mechanical energy conservation most often involves the coefficient of restitution.[5] The duration of the impact process is considered to be infinitesimally small, such that the form assumed by the contact forces is that of a Dirac-delta function. This introduces a limitation whereby the external forces are not included in the analysis; only the contact forces are considered to significantly influence the impact dynamics. Further curbing the method's accuracy and real-time potential is its being non-linear in slip-impact situations, and the fact that time integration of the equations is discontinuous across the impact, requiring that the system be re-initialized. (Remark that slip, or relative motion between the bodies in the plane tangent to the contact surfaces, may be present both in the impact and the continuous contact cases.)

The approach presented in this paper borrows from both previous methods. It is time continuous, which therefore allows for conducting the analysis of the system by simply adding the contact forces to the equations of motion. The time step, however, remains constant throughout the simulation, an advantage which is not shared by the continuous time method. The contact forces are calculated by considering the impulse balance principle. The latter, however, is augmented in such a way as to fully take into account the external forces acting on the bodies. This allows for more accurately handling the impact cases and, more importantly, makes it possible to model continuous contact in the most simple form possible. Global momentum conservation is enforced during contact, subject to a form of energy conservation involving the coefficient of restitution. The method is not iterative, and remains linear even in slip-contact situations by appropriately defining the slip direction.

To demonstrate the applicability of the method, a few simple impact and continuous contact cases will be analyzed using the direct central impact benchmark. The full capabilities of the method will be highlighted by a system-level functional simulation of the European Automated Transfer Vehicle (ATV) docking with the International Space Station using the CAE ROSE (Real-time Object-oriented Simulation Environment). Included in the simulation are the ATV attitude control and propulsion subsystems, and the ATV/ISSA orbital dynamics and environmental models. Study of the collision between the ISSA and ATV following misfiring of faulty thrusters is considered.

Theory

Notations

In keeping with standard terminology[6] relating to rendezvous operations of space vehicles the two colliding bodies will be identified as the 'target' and 'chaser', respectively; subscripts '*t*' and '*c*' affixed to variables will be used to denote them. Vectors and dyadics will be written in plain bold type using arrows, such as $\vec{\mathbf{v}}_t$ or $\vec{\mathbf{I}}_t$, and a left superscript will replace the arrow when the vector or dyadic is expressed in a specific frame of reference. For example, $\vec{\mathbf{v}}_t$ expressed in the geocentric inertial frame will be denoted by ${}^{g}\mathbf{v}_t$, and $\vec{\mathbf{I}}_t$ expressed in the target center-of-mass (COM) frame by ${}^{t}\mathbf{I}_t$. Four reference frames will be used throughout. These are the geocentric inertial, target COM, chaser COM, and contact frames, denoted by left superscripts '*g*', '*t*', '*c*', and '*k*', respectively; those frames are defined below. Subscripts '*x*', '*y*', and '*z*' will identify a vector's Cartesian components, and superscript '*T*' denote the transpose of a vector or matrix.

Equations of motion

The target and chaser each have a reference frame attached to their center-of-mass; these are called the target COM frame and chaser COM frame, respectively, and are shown in Figure 1 as '*t*' and '*c*'. Also shown in the figure is the contact frame of reference '*k*', defined on the target at the point of contact. Its *z*-axis is outward pointing and normal to the surface. The position vector $\vec{\mathbf{r}}_{k/t}$ locates the point of contact on the target relative to the target center-of-mass, and, similarly, the vector $\vec{\mathbf{r}}_{k/c}$ locates the point of contact on the chaser.

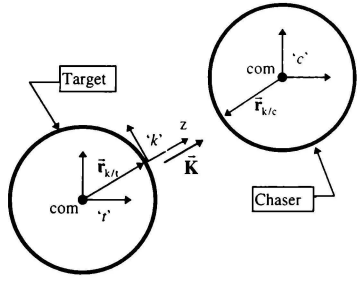


Fig 1 Target and chaser, with definition of contact frame, '*k*'.

The contact force, denoted by $\vec{\mathbf{K}}$, is defined as the force applied by the target on the chaser; its normal component is thus always positive when expressed in the contact frame.

The equations of motion, including the contact force $\vec{\mathbf{K}}$, the externally applied forces, $\vec{\mathbf{F}}_t$ and $\vec{\mathbf{F}}_c$, and the externally applied moments, $\vec{\mathbf{G}}_t$ and $\vec{\mathbf{G}}_c$, are given by

$$m_t \frac{{}^{\kappa}d\vec{\mathbf{v}}_t}{dt} = \vec{\mathbf{F}}_t - \vec{\mathbf{K}} \tag{1}$$

$$\vec{\mathbf{I}}_t \cdot \frac{{}^{t}d\vec{\boldsymbol{\omega}}_t}{dt} = \vec{\mathbf{G}}_t - \vec{\boldsymbol{\omega}}_t \times \left(\vec{\mathbf{I}}_t \cdot \vec{\boldsymbol{\omega}}_t\right) - \vec{\mathbf{r}}_{k/t} \times \vec{\mathbf{K}} \tag{2}$$

$$m_c \frac{{}^{\kappa}d\vec{\mathbf{v}}_c}{dt} = \vec{\mathbf{F}}_c + \vec{\mathbf{K}} \tag{3}$$

$$\vec{\mathbf{I}}_c \cdot \frac{{}^{c}d\vec{\boldsymbol{\omega}}_c}{dt} = \vec{\mathbf{G}}_c - \vec{\boldsymbol{\omega}}_c \times \left(\vec{\mathbf{I}}_c \cdot \vec{\boldsymbol{\omega}}_c\right) + \vec{\mathbf{r}}_{k/c} \times \vec{\mathbf{K}} \tag{4}$$

In the above m is the mass, $\vec{\mathbf{v}}$ the velocity vector, $\vec{\mathbf{I}}$ the inertia dyadic calculated about the mass center, and $\vec{\boldsymbol{\omega}}$ the angular velocity vector. Whereas the time derivative in Eqs. (1) and (3) is measured in the inertial frame, it is convenient to measure it in the target COM frame in Eq. (2), and in the chaser COM frame in Eq. (4); the left superscript on the time derivatives indicates that notation.

In the introduction we mentioned that the coefficient of restitution and contact force both affect the system post-impact configuration. Because there is always rather great uncertainty associated with estimating the coefficient of restitution[4] (or the modal damping coefficients), some inaccuracy will be introduced in the post-impact configuration which will defeat any effort made for accurately modeling the profile of the contact force. This suggests that it is possible to relax the precision with which the latter is calculated. We exploit that by first imposing on the collision the constraint that its duration is equal to Δt seconds, the time step used for integrating the equations numerically. Furthermore, the contact force is assumed to remain constant during this time interval:

$$\vec{\mathbf{K}} = \text{cst} \neq 0, \quad t \in \left[t^n,\ t^n + \Delta t\right] \tag{5}$$

and

$$\vec{\mathbf{K}} = 0, \quad t \notin \left[t^n,\ t^n + \Delta t\right] \tag{6}$$

$t^n = n\Delta t$ denotes the time level reached after n time steps. By scheduling the contact dynamics modules in the real-time simulation such that their period has the duration of a typical collision (20–40 ms), a balance will be struck between the inaccuracies associated with the contact force and coefficient of restitution. It is important to realize, however, that momentum conservation still and always will be enforced, as detailed below, and that this only affects the system post-collision configuration. Moreover, any error resulting from the profile assumed for the contact force will be of order Δt, which is small.

Applying the following operator to the equations of motion (1)-(4):

$$\int_{t^n}^{t^{n+1}} (\ldots)\, dt \tag{7}$$

we obtain

$$m_t \vec{\mathbf{v}}_t^{+} - m_t \vec{\mathbf{v}}_t^{-} = \vec{\mathbf{f}}_t - \vec{\mathbf{k}} \tag{8}$$

$$\vec{\mathbf{I}}_t \cdot \vec{\boldsymbol{\omega}}_t^{+} - \vec{\mathbf{I}}_t \cdot \vec{\boldsymbol{\omega}}_t^{-} = \vec{\mathbf{g}}_t - \vec{\mathbf{g}}_t' - \vec{\mathbf{r}}_{k/t} \times \vec{\mathbf{k}} \tag{9}$$

$$m_c \vec{\mathbf{v}}_c^{+} - m_c \vec{\mathbf{v}}_c^{-} = \vec{\mathbf{f}}_c + \vec{\mathbf{k}} \tag{10}$$

$$\vec{\mathbf{I}}_c \cdot \vec{\boldsymbol{\omega}}_c^{+} - \vec{\mathbf{I}}_c \cdot \vec{\boldsymbol{\omega}}_c^{-} = \vec{\mathbf{g}}_c - \vec{\mathbf{g}}_c' + \vec{\mathbf{r}}_{k/c} \times \vec{\mathbf{k}} \tag{11}$$

where superscript '-' indicates that the solution is at time t^n, before the collision, and superscript '+' at time t^{n+1}, after the collision. By making the assumption in Eqs. (1)–(4) that the terms on the right-hand side are constant in the interval $\left[t^n, t^{n+1}\right]$, the impulses of the forces and moments in Eqs. (8)–(11) are given by

$$\begin{aligned} &\vec{\mathbf{k}} = \Delta t\, \vec{\mathbf{K}}, \\ &\vec{\mathbf{f}}_t = \Delta t\, \vec{\mathbf{F}}_t, \quad \vec{\mathbf{g}}_t = \Delta t\, \vec{\mathbf{G}}_t, \quad \vec{\mathbf{g}}_t' = \Delta t\, \vec{\boldsymbol{\omega}}_t \times \left(\vec{\mathbf{I}}_t \cdot \boldsymbol{\omega}_t\right) \\ &\vec{\mathbf{f}}_c = \Delta t\, \vec{\mathbf{F}}_c, \quad \vec{\mathbf{g}}_c = \Delta t\, \vec{\mathbf{G}}_c, \quad \vec{\mathbf{g}}_c' = \Delta t\, \vec{\boldsymbol{\omega}}_c \times \left(\vec{\mathbf{I}}_c \cdot \boldsymbol{\omega}_c\right) \end{aligned} \tag{12}$$

Equations (8) and (10) can now be written in the inertial frame, Eq. (9) in the target COM frame, and Eq. (11) in chaser COM frame, to obtain

$$m_t\, {}^{\kappa}\mathbf{v}_t^{+} + \left({}^{k}\mathbf{M}^{\kappa}\right)^{T} {}^{k}\mathbf{k} = {}^{\kappa}\mathbf{f}_t + m_t\, {}^{\kappa}\mathbf{v}_t^{-} \tag{13}$$

$${}^{t}\mathbf{I}_t\, {}^{t}\boldsymbol{\omega}_t^{+} - \left({}^{k}\mathbf{M}^{t}\, {}^{t}\mathbf{r}_{k/t}^{\times}\right)^{T} {}^{k}\mathbf{k} = {}^{t}\mathbf{g}_t - {}^{t}\mathbf{g}_t' + {}^{t}\mathbf{I}_t\, {}^{t}\boldsymbol{\omega}_t^{-} \tag{14}$$

$$m_c\, {}^{\kappa}\mathbf{v}_c^{+} - \left({}^{k}\mathbf{M}^{\kappa}\right)^{T} {}^{k}\mathbf{k} = {}^{\kappa}\mathbf{f}_c + m_c\, {}^{\kappa}\mathbf{v}_c^{-} \tag{15}$$

$${}^{c}\mathbf{I}_c\, {}^{c}\boldsymbol{\omega}_c^{+} + \left({}^{k}\mathbf{M}^{c}\, {}^{c}\mathbf{r}_{k/c}^{\times}\right)^{T} {}^{k}\mathbf{k} = {}^{c}\mathbf{g}_c - {}^{c}\mathbf{g}_c' + {}^{c}\mathbf{I}_c\, {}^{c}\boldsymbol{\omega}_c^{-} \tag{16}$$

where the right superscript '×' means that the corresponding vector is formed into the skew-symmetric matrix used for performing the cross vector product. For example, given a vector $\mathbf{a} = \begin{bmatrix} a_x & a_y & a_z \end{bmatrix}^T$, we have

$$\mathbf{a}^{\times} = \begin{bmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{bmatrix} \tag{17}$$

In the above, $\mathbf{M}$ denotes the rotation matrix used for transforming vector quantities between coordinate frames; for example, pre-multiplication of a vector by ${}^{k}\mathbf{M}^{t}$ transforms its coordinates from the target COM frame to the contact frame. Similar definitions apply for the other matrices, whereby the right and left superscripts affixed to the matrix indicate the direction of the transformation. Remark that we elect to express the impulse of the contact force, ${}^{k}\mathbf{k}$, in the contact frame.

The system of equations (13)–(16) consists of 12 equations (4 vector equations), with the number of unknowns being greater than 12. The kinematics constraints on the surfaces at the point of contact provide the additional relations required to close the system. Two cases are of interest, depending on whether or not there is slip between the contact surfaces during the collision. The next two subsections separately deal with these two cases.

No-Slip Contact

It is not possible to know beforehand whether the contact surfaces will experience slip during the collision, or indeed whether friction will be strong enough to prevent relative tangential motion between them. Most contact dynamics solution procedures resort to iterative approaches that are often non-linear to resolve this issue (especially in slip-contact situations).[5] Since those are not desirable features for a method which is designed to run in real-time, we propose an approach that is linear and non-iterative; it consists of two steps.

In the first step, the solution proceeds by always assuming that there is *no slip* between the contact surfaces of the target and chaser. After calculating the contact force subject to that assumption, we check whether or not the underlying no-slip condition is satisfied. If it is satisfied, then the solution is consistent and the solution procedure may be terminated. In case to the contrary, a slip-contact solution must be computed in a second step, where use is made of information carried over from the first step.

Thus, we must first define, as part of the first step, the kinematics constraint governing no-slip contact. It relates the relative velocity of approach and of separation of the contact points (the normal, or z-component when expressed in the contact frame).[4] Their ratio is equal to the coefficient of restitution, e. Furthermore, the relative velocity between the contact points in the planes tangent to the surfaces is equal to zero. Thus, we can express this mathematically as

$$\begin{aligned} &\left({}^{k}\mathbf{M}^{\kappa}\,{}^{\kappa}\mathbf{v}_t^{+} + {}^{k}\mathbf{M}^{t}\,{}^{t}\mathbf{r}_{k/t}^{\times}\,{}^{t}\boldsymbol{\omega}_t^{+}\right) - \\ &\left({}^{k}\mathbf{M}^{\kappa}\,{}^{\kappa}\mathbf{v}_c^{+} + {}^{k}\mathbf{M}^{c}\,{}^{c}\mathbf{r}_{k/c}^{\times}\,{}^{c}\boldsymbol{\omega}_c^{+}\right) = \\ &\mathbf{E}\left({}^{k}\mathbf{V}_t^{-} - {}^{k}\mathbf{V}_c^{-}\right) \end{aligned} \tag{18}$$

where

$$\begin{aligned} {}^{k}\mathbf{V}_t^{-} &= \left({}^{k}\mathbf{M}^{\kappa}\,{}^{\kappa}\mathbf{v}_t^{-} + {}^{k}\mathbf{M}^{t}\,{}^{t}\mathbf{r}_{k/t}^{\times}\,{}^{t}\boldsymbol{\omega}_t^{-}\right) \\ {}^{k}\mathbf{V}_c^{-} &= \left({}^{k}\mathbf{M}^{\kappa}\,{}^{\kappa}\mathbf{v}_c^{-} + {}^{k}\mathbf{M}^{c}\,{}^{c}\mathbf{r}_{k/c}^{\times}\,{}^{c}\boldsymbol{\omega}_c^{-}\right) \\ \mathbf{E} &= \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & -e \end{bmatrix} \end{aligned} \tag{19}$$

The coefficient of restitution, e, in matrix $\mathbf{E}$ above represents the degree of elasticity of the impact[4]: $e = 1$ represents a completely elastic collision, and $e = 0$ a completely plastic one; it must satisfy $0 \le e \le 1$.

Equations (13)–(16) and (18) now form a complete set which can be solved to obtain the 15 unknowns ${}^{\kappa}\mathbf{v}_t^{+}$, ${}^{t}\boldsymbol{\omega}_t^{+}$, ${}^{\kappa}\mathbf{v}_c^{+}$, ${}^{c}\boldsymbol{\omega}_c^{+}$, ${}^{k}\mathbf{k}$. It is to be remarked, however, that only the impulse of the contact force, ${}^{k}\mathbf{k}$, is of interest because of the time continuous nature of the method. It can be obtained by eliminating the remaining variables from the system of equations, and is given by

$${}^{k}\mathbf{k} = \mathbf{A}^{-1}\left(\mathbf{b}_2 - \mathbf{T}\mathbf{b}_1\right) \tag{20}$$

where

$$\begin{aligned} \mathbf{A} = &-m_t^{-1}\mathbf{1} - \left\{{}^{k}\mathbf{M}^{t}\,{}^{t}\mathbf{r}_{k/t}^{\times}\right\}{}^{t}\mathbf{I}_t^{-1}\left\{{}^{k}\mathbf{M}^{t}\,{}^{t}\mathbf{r}_{k/t}^{\times}\right\}^{T} \\ &-m_c^{-1}\mathbf{1} - \left\{{}^{k}\mathbf{M}^{c}\,{}^{c}\mathbf{r}_{k/c}^{\times}\right\}{}^{c}\mathbf{I}_c^{-1}\left\{{}^{k}\mathbf{M}^{c}\,{}^{c}\mathbf{r}_{k/c}^{\times}\right\}^{T} \end{aligned} \tag{21}$$

$$\begin{aligned} \mathbf{T} = &\left[m_t^{-1}\,{}^{k}\mathbf{M}^{\kappa} \quad -\left\{{}^{k}\mathbf{M}^{t}\,{}^{t}\mathbf{r}_{k/t}^{\times}\right\}{}^{t}\mathbf{I}_t^{-1}\right. \\ &\left.-m_c^{-1}\,{}^{k}\mathbf{M}^{\kappa} \quad \left\{{}^{k}\mathbf{M}^{c}\,{}^{c}\mathbf{r}_{k/c}^{\times}\right\}{}^{c}\mathbf{I}_c^{-1}\right] \end{aligned} \tag{22}$$

$$\mathbf{b}_1 = \begin{Bmatrix} {}^{\kappa}\mathbf{f}_t + m_t\,{}^{\kappa}\mathbf{v}_t^{-} \\ {}^{t}\mathbf{g}_t - {}^{t}\mathbf{g}_t^{I} + {}^{t}\mathbf{I}_t\,{}^{t}\boldsymbol{\omega}_t^{-} \\ {}^{\kappa}\mathbf{f}_c + m_c\,{}^{\kappa}\mathbf{v}_c^{-} \\ {}^{c}\mathbf{g}_c - {}^{c}\mathbf{g}_c^{I} + {}^{c}\mathbf{I}_c\,{}^{c}\boldsymbol{\omega}_c^{-} \end{Bmatrix} \tag{23}$$

$$\mathbf{b}_2 = \mathbf{E}\left({}^{k}\mathbf{V}_t^{-} - {}^{k}\mathbf{V}_c^{-}\right) \tag{24}$$

In the above, $\mathbf{A}$ is a 3×3 matrix, $\mathbf{T}$ a 3×12 matrix composed of four 3×3 blocks, $\mathbf{b}_1$ a 12×1 column vector, and $\mathbf{b}_2$ a 3×1 column vector; $\mathbf{1}$ denotes the 3×3 unit matrix. Thus, it is seen that it is only necessary to invert a 3×3 matrix to solve for the contact force impulse.

Now, it must be verified that the impulse of the contact force, ${}^k\mathbf{k} = \begin{bmatrix} k_x & k_y & k_z \end{bmatrix}^T$, satisfies the no-slip condition:

$$\sqrt{k_x^2 + k_y^2} < \mu_s k_z \tag{25}$$

where μ_s is the coefficient of *static* friction. In case it is affirmative, ${}^k\mathbf{k}$ represents a consistent solution and the contact forces and contact moments on the target and chaser may be calculated from Eqs. (34)–(37) below. If condition (25) is violated, a slip contact solution must be obtained instead, as described in the next subsection.

Slip Contact

The contact force consists of a normal and a tangential component, respectively along the *z*-axis and in the *x*-*y* plane of the contact frame. In slip contact situations the tangential component is directed along the tangential projection of the vector describing the relative velocity between the contact points of the target and chaser[5]. More simply, we have, mathematically,

$${}^k\mathbf{k} = \begin{bmatrix} \mu u_x \\ \mu u_y \\ 1 \end{bmatrix} k_z \equiv {}^k\mathbf{U}\, k_z \tag{26}$$

where

$$u_x = \frac{\left({}^k\mathbf{V}_t^- - {}^k\mathbf{V}_c^-\right)_x}{\left\| {}^k\mathbf{V}_t^- - {}^k\mathbf{V}_c^- \right\|_\Sigma}, \qquad u_y = \frac{\left({}^k\mathbf{V}_t^- - {}^k\mathbf{V}_c^-\right)_y}{\left\| {}^k\mathbf{V}_t^- - {}^k\mathbf{V}_c^- \right\|_\Sigma}, \tag{27}$$

and

$$\left\| {}^k\mathbf{V}_t^- - {}^k\mathbf{V}_c^- \right\|_\Sigma = \sqrt{\left({}^k\mathbf{V}_t^- - {}^k\mathbf{V}_c^-\right)_x^2 + \left({}^k\mathbf{V}_t^- - {}^k\mathbf{V}_c^-\right)_y^2} \tag{28}$$

${}^k\mathbf{V}_t^-$ and ${}^k\mathbf{V}_c^-$, the pre-impact velocity of the target and chaser, are given in Eq. (19), and μ represents the coefficient of *dynamic* friction. Remember that k_z is always positive when expressed in the contact frame, being the force applied by the target on the chaser. Also, notice the definition for ${}^k\mathbf{U}$ in Eq. (26).

Methods inspired by that of Kane and Levinson[5] for handling slip contact are usually iterative and make use of the post-impact velocity, ${}^k\mathbf{V}_t^+$ and ${}^k\mathbf{V}_c^+$, instead of the pre-impact velocity [as in Eq. (27)]. Both approaches are equivalent, at least when the pre-contact relative velocity, ${}^k\mathbf{V}_t^- - {}^k\mathbf{V}_c^-$, is different from zero, with the present method having the advantage of not being iterative. When ${}^k\mathbf{V}_t^- - {}^k\mathbf{V}_c^-$ is equal to zero, however, for example when both bodies are at rest and touch each other and a force is applied on either of them to initiate motion, the present method breaks down (using the post-impact velocities has the effect of circumventing that problem). We now show that it is possible to modify the present formulation in order to accommodate that singular condition.

The contact force calculated in Eq. (20) during the first step had a tangential component which reflected the influence of the external and momentum forces on the dynamics. It is therefore reasonable to assume that, when the pre-impact relative velocity is equal to zero and slip occurs, the slip direction will be aligned with the tangential component of the contact force. Thus, we use the following expressions for u_x and u_y when ${}^k\mathbf{V}_t^- - {}^k\mathbf{V}_c^-$ is equal to zero:

$$u_x = \frac{k_x}{\left\| {}^k\mathbf{k} \right\|_\Sigma}, \qquad u_y = \frac{k_y}{\left\| {}^k\mathbf{k} \right\|_\Sigma}, \tag{29}$$

where

$$\left\| {}^k\mathbf{k} \right\|_\Sigma = \sqrt{k_x^2 + k_y^2} \tag{30}$$

${}^k\mathbf{k}$ was obtained in Eq. (20), and all conditions can now be accommodated.

It is seen from Eq. (26) that there is only one unknown in the slip contact situation, namely k_z, the normal component of the impulse of the contact force. Only one additional equation is therefore required to close the system of equations. It is provided by the *z*-component of Eq. (18). After substituting Eq. (26) into the equations of motion and solving for k_z, the solution is readily shown to be equal to

$$k_z = \frac{1}{a}\left(\mathbf{b}_2 - \mathbf{T}\mathbf{b}_1\right)_z \tag{31}$$

where

$$a = \mathbf{A}_z\, {}^k\mathbf{U} \tag{32}$$

Eq. (32) represents the scalar product of vector ${}^k\mathbf{U}$, defined in Eq. (26), and the third row of matrix $\mathbf{A}$ given by Eq. (21), whereas $\mathbf{T}$, $\mathbf{b}_1$ and $\mathbf{b}_2$ are defined by Eqs. (22)–(24).

Contact Forces and Moments

In component form, the contact force, ${}^k\mathbf{K}$, is related to the contact impulse, ${}^k\mathbf{k}$, by the following relation [see equation (12)]:

$${}^k\mathbf{K} = \frac{1}{\Delta t}\, {}^k\mathbf{k} \tag{33}$$

The contact forces and contact moments applied on the target and chaser that are to be added to the equations of motion after detecting a collision are given by

$$ {}^{\kappa}\mathbf{F}_t^k = -\left({}^k\mathbf{M}^{\kappa}\right)^T {}^k\mathbf{K} \tag{34} $$

$$ {}^{\prime}\mathbf{M}_t^k = \left({}^k\mathbf{M}^{\prime}\,{}^{\prime}\mathbf{r}_{k/t}^{\kappa}\right)^T {}^k\mathbf{K} \tag{35} $$

$$ {}^{\kappa}\mathbf{F}_c^k = \left({}^k\mathbf{M}^{\kappa}\right)^T {}^k\mathbf{K} \tag{36} $$

$$ {}^{\prime}\mathbf{M}_c^k = -\left({}^k\mathbf{M}^{\prime}\,{}^{\prime}\mathbf{r}_{k/c}^{\kappa}\right)^T {}^k\mathbf{K} \tag{37} $$

Direct Central Impact

To establish the viability of the continuous impulse balance method it is first applied to the direct central impact problem, where distinction is made between the free motion and forced motion cases. In the free motion case, no external forces are applied on either body during the collision; this is purely an impact problem, with only contact forces and contact moments influencing the dynamics. Forced motion, on the other hand, allows for both external and contact forces to intervene, and a combination of impact and continuous contact situations can be analyzed. Remark that slip-contact will never occur in this problem, and therefore the actual values for the coefficients of friction μ_s and μ are not relevant; they have been set to 0.6 and 0.4, respectively.

Free Motion

The free motion direct central impact problem has an analytical solution, obtained by considering momentum and energy conservation principles.[4] It is representative of the nature of the problem to assume that the bodies undergoing collision are spherical. The radius vector joining the spheres' mass centers may be taken to be aligned, say, with the x-axis of an inertial frame of reference, with motion being likewise along the x-axis. Considering that both bodies are of equal mass, the analytical solution is given by

$$ v_t^+ = \frac{1+e}{2} v_c^- \tag{38} $$

$$ v_c^+ = \frac{1-e}{2} v_c^- \tag{39} $$

where v represents velocity. x_t and x_c will denote the position of the target and chaser mass centers, respectively, and we assume that $x_t > x_c$ (the target is in front of the chaser).

Integration of the equations of motion (1)–(4) to obtain a numerical solution to the direct central impact problem for comparison with Eqs. (38) and (39) will be performed using the Runge-Kutta fourth-order scheme (except where mentioned). To determine whether collision is imminent the distance between the two bodies, Δx, must be determined as follows:

$$ \Delta x = \left(x_t - a_t\right) - \left(x_c + a_c\right) \tag{40} $$

We consider that collision will take place during the next time step when $\Delta x \le 0$; the contact forces and contact moments acting on the target and chaser are then calculated and added to the equations of motion. Contact detection—the calculation of Δx—is performed every time step.

The present examples all involve two bodies that have identical mass and radius, respectively $m_t = m_c =$ 1 kg, and $a_t = a_c =$ 1 m. The target is initially at rest, with the chaser moving toward it on a collision course with a velocity of 1 m/s. Table 1 summarizes the quantities of interest. The x-position of the bodies right at the moment collision was detected, and on the time step immediately following collision, is tabulated, along with the x-velocity. (The initial angular velocity of the bodies has been set equal to zero and therefore remains so throughout the collision.) The table gives results for three coefficients of restitution, e=0, e=1, and e=0.5.

It can be verified that the numerical post-impact velocities are always in perfect agreement with the analytical results for all values of e (momentum conservation is always enforced). As mentioned in the introduction, varying the coefficient of restitution has the effect of changing the system post-impact configuration. Thus, the post-impact velocity of either body strongly depends on e, as well as the distance traveled during

		Coefficient of restitution, e		
		0	0.5	1
Before collision	x_c^-	1	1	1
	x_t^-	3	3	3
	$\dot{x}_c^-$	1	1	1
	$\dot{x}_t^-$	0	0	0
After collision	x_c^+	1.075	1.0625	1.025
	x_t^+	3.025	3.0375	3.025
	$\dot{x}_c^+$	0.5	0.25	0
	$\dot{x}_t^+$	0.5	0.75	1
	Δx	-0.05	-0.025	0

Table 1 Free motion direct central impact for two bodies of equal mass.

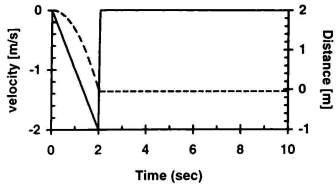


Fig 2 Forced motion direct central impact, e=0. —, velocity; ---, displacement.

the collision. Because the initial conditions have been set such that collision is detected exactly when $\Delta x = 0$, the interference between the bodies on the time step immediately following collision constitutes a good measure of the mechanical energy lost during the impact. It is seen that there is interference, or material penetration of the surfaces ($\Delta x < 0$), when e<1, as expected.

The time step in the above calculations was set equal to 0.05 second. Results obtained by changing the time step to 0.025 second, for example, instead of 0.05 second, have shown that the effect is to correspondingly halve all of the interference values (and double the magnitude of the contact force). The post-impact velocities, however, remained the same because the method enforces momentum conservation.

Let's now briefly digress to report on calculations performed using the following two additional time integration schemes: (1) fourth-order predictor corrector Adams-Moulton (AM4), and (2) first-order Euler. AM4, while having the same formal order of accuracy as Runge-Kutta, does not qualify as an effective alternative to it. Indeed, because of its use of values from three previous time steps it does not allow for transferring momentum between the bodies in the single time step during which the contact forces are applied. The

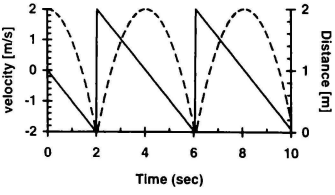


Fig 3 Forced motion direct central impact, e=1. —, velocity; ---, displacement.

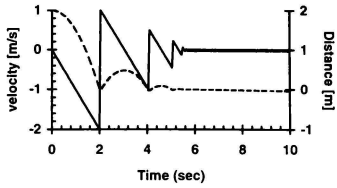


Fig 4 Forced motion direct central impact, e=0.5. —, velocity; ---, displacement.

Euler scheme, although of lower accuracy, can effect the desired momentum transfer in the required time interval. Subsequent calculations all make use of the fourth-order Runge-Kutta scheme.

Forced Motion

Forced motion provides for a much more interesting benchmark with which to investigate not only impact, but also continuous contact. We consider the same system as in the previous free motion example, with the difference that both bodies are initially at rest. A constant force of 1 N is applied on the chaser throughout the simulation such as to induce direct central motion toward to target; no force is applied on the target. Plots of the distance between the target and chaser, Δx, and of their relative velocity, $v_t - v_c$, will be used to highlight the system response. The distance, Δx, between the target and chaser in initially set to 2 m.

In Figure 2 the coefficient of restitution is equal to zero (e=0). It is seen that the target and chaser move as a single body after the collision; distance and relative velocity between them are equal to zero. Their acceleration, calculated by differencing the numerical velocity, is exactly equal to 0.5 m/s^2. This is in full agreement with the analytical solution, $F_c/(m_c + m_t)$, where $F_c = 1$ is the force applied on the chaser.

Results for the other extreme case where e=1 are presented in Figure 3. The target bounces off the chaser after the collision with a constant velocity. However, the force still being applied on the chaser, the latter catches up with the target, thus initiating a series of collision cycles that repeats itself indefinitely. This is in contrast to the results of Figure 4 where e=0.5. In this case, some energy is lost during the impact with each successive collision. The distance the target bounces off the chaser and the relative velocity between them become progressively smaller. Whereas in the free motion case the initial conditions had been set such that collision always occurred when $\Delta x = 0$, collision in

general will be detected when $\Delta t|v_i - v_j| < \Delta x \le 0$ because of the discrete nature of the problem. The effect of this is illustrated in Figure 4 where it is seen that the interference tends to increase with time, albeit very slowly. To counter that effect we can set ϵ=0 after detecting that $\Delta x < 2|v_i - v_j|\Delta t$, for example; calculations not presented here have shown that results in this case tend asymptotically to those of Figure 2.

Docking of ATV and Space Station

The Automated Transfer Vehicle (ATV) is part of the European Space Agency's (ESA) contribution to the International Space Station (ISS). Although the ATV's detailed mission profile still is in the definition stage, agreed upon mission scenarios include the re-supply and re-boosting of the ISS. Mission plans call for the Ariane 5 rocket to inject the ATV into low Earth orbit, following which autonomous propulsion and control capabilities will allow the ATV to effect the orbital transfer maneuvers necessary to come within proximity of and to dock to the ISS. Such typical mission phases as injection, phasing and rendezvous have been implemented in a full functional system level simulation using the CAE ROSE (Real-time Object-oriented Simulation Environment). Included in the simulation are the ATV propulsion and attitude control subsystems, ATV on-board equipment and on-board software (for planning and executing the mission), and the ATV/ISS orbital dynamics models; a visual renderer makes it possible to visualize the maneuvers in real-time.

For the purpose of demonstrating the ability of the

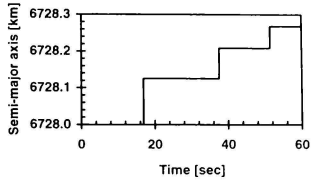


Fig 6 Semi-major axis of ISS as a function of time following each successive collision.

continuous impulse balance method to handle complex contact dynamics problems, we will concentrate on the docking phase of the mission, that which terminates the rendezvous maneuver. We define the ISS and ATV, shown in Figure 5, to be the target and chaser, respectively. The ISS is modeled as a single rigid orbiting body the total mass of which is m_t=217480 kg. Integration of the ISS equation of motion in translation, Eq. (1), proceeds by considering that, apart from the contact forces, only gravitational forces affect its dynamics; gravitational acceleration includes effects from the J2, J3, J4 zonal and J22 tesseral harmonics. Angular motion of the space station is prescribed such that its z-axis stays Earth-pointing; this allows for dropping Eq. (2) from the set of equations to be integrated. The contact dynamics formulation handles this by setting ${}^t\mathbf{I}_t^{-1}$ equal to zero in Eqs. (21) and (22) for **A** and **T**.

Both equations (3) and (4) for translation and angular motion of the ATV are integrated simultaneously with

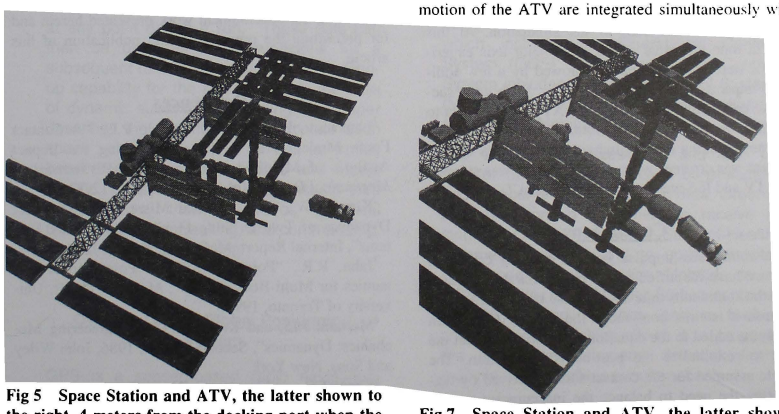


Fig 5 Space Station and ATV, the latter shown to the right, 4 meters from the docking port when the thrusters were failed 'On'.

Fig 7 Space Station and ATV, the latter shown after the third collision had occurred.

Eq. (1). The mass of the ATV is equal to 17594 kg, and its inertia matrix is given by

$$ {}^{c}\mathbf{I}_{c} = \begin{bmatrix} 40558 & 0 & 0 \\ 0 & 216755 & 0 \\ 0 & 0 & 216628 \end{bmatrix} \mathrm{kg \cdot m^2} \qquad (41) $$

Determination of the ATV angular motion requires that Eq. (4) be augmented by equations describing the time rate of change of the attitude; relations involving the Euler parameters were used for that purpose. Insofar as the ATV is concerned, control forces from the propulsion subsystem must be considered in addition to the gravitational and contact forces.

The nominal docking scenario that we consider has the ATV coming in from behind the ISS along the V-Bar Approach Corridor and catching up with it at a rate of 2 cm/s. To clearly highlight the capabilities of the contact dynamics method, we will simulate a much degraded, if improbable, operations scenario wherein all attitude control thrusters suddenly fail 'On' when the ATV is to within 4 meters of the ISS. This has the effect of propelling the ATV toward the ISS with a net thrust force of approximately 1600 N. We will take the coefficient of restitution to be equal to e=0.7, with the coefficients of friction being $\mu_s = 0.6$ and $\mu = 0.4$.

Plots of the distance and relative velocity between the ATV and ISS would look very much like those of the previous section. It is for that reason that we will rather highlight the effect that the collision has on the ISS orbital parameters, more specifically on its semi-major axis.[6] Thus, as seen in Figure 6, the latter stays constant until the occurrence of the first collision, approximately 17 seconds after the start of the simulation. At this point in time the value of the semi-major axis experiences a step increase. This is followed by a few additional impacts of decreasing intensity. With each successive collision the attitude of the ATV with respect to the ISS changed because the contact point on the ATV does not lie along its longitudinal x-axis, and a contact moment was created; when the simulation was stopped the ATV and ISS were as shown in Figure 7.

Conclusions

The continuous impulse balance method combines what we have identified as the most desirable features from the continuous time and impulse balance methods. Because it is time continuous the contact forces can simply be added to the equations of motion without the need to re-initialize the post-collision solution. The profile assumed for the contact forces, derived by recognizing the need to balance the inaccuracies associated with evaluating non-conservative effects, makes it possible to keep the time step constant during integration of the equations. This facilitates the real-time implementation of the method, and does not impose the computational burdens otherwise associated with refining the time step. Explicit expressions for the contact forces are obtained by enforcing momentum conservation on the system—and it is only necessary to inverse a 3×3 matrix to solve for them. The formulation accounts for the external forces and therefore makes it possible to handle both impact and continuous contact in a completely seamless fashion. Continuous contact, in particular, does not require that an iterative solution procedure be adopted. The same applies to slip-contact situations where the method has the additional benefit of maintaining linearity. The latter is achieved by appropriately defining the slip direction using the pre-impact velocity or, when the latter is equal to zero, information derived from preliminary no-slip contact force calculations.

The benchmark direct central impact problem demonstrated the ability of the continuous impulse balance method to accurately deal with impact and continuous contact. Further, it highlighted the role of the coefficient of restitution (non-conservative forces) and contact force profile (in our case influenced by the time step) in determining the system post-impact configuration. Incorporation of the method within a full system level functional simulation of the ATV and ISS proved to be an easy and straightforward task, and the results obtained established its real-time potentials.

Acknowledgments

We wish to acknowledge CAE Electronics Ltd. for supporting the development work presented herein and for permitting the preparation and publication of this article.

References

[1]Lankarani, H.M., and Nikravesh, P.E., "A Contact Force Model With Hysteresis Damping for Impact Analysis of Multibody Systems", ASME *Journal of Mechanical Design* Vol. 112, 1990, pp. 112-376.

[2]Kim, S.-W., Cyril, X., and Misra, A.K., "Contact Dynamics of Two Orbiting Flexible Multi-Body Systems", Internal Report, McGill University, 1995.

[3]Jahn, R.R., "Point-Contact and Point-Impact Dynamics for Multi-Body Chains", *Master's Thesis*, University of Toronto, 1993.

[4]Meriam, J.L., and Kraige, L.G., "Engineering Mechanics: Dynamics", Second Edition, 1986, John Wiley and Sons, New York.

[5]Kane, T.R., and Levinson, D.A., "Dynamics: Theory and Applications", McGraw-Hill, New York, 1985.

[6]Griffin, D.G., and French, J.R, "Space Vehicle Design", AIAA Education Series, 1991.

HIGH FIDELITY GPS SATELLITE SIMULATION

Mark Baker, Lockheed Martin Federal Systems - Gaithersburg
Brad Dingman, Lockheed Martin Federal Systems - Gaithersburg
William Gregg, Lockheed Martin Federal Systems - Gaithersburg

Abstract

Lockheed Martin Federal Systems is developing a high fidelity simulator for the Air Force's Global Positioning System (GPS) satellite command and control operations. This paper discusses the system architecture of the high fidelity GPS simulator, the handling of the state data for a simulation, and lastly, the space vehicle subsystem dynamic modeling algorithms. The system architecture consists primarily of a symmetric multiprocessor performing processor intensive functions of Space and Environment Simulation while a set of distributed workstation processors implement an assortment of user and communications interfaces. Simulation runs use a large set of state data pertaining to both the space vehicles and ground components. Users are able to manipulate a set of initial conditions (state data) before a run, and are able to save sets of state data at any point during a run. These sets can then be used as initial conditions for a subsequent run, in effect allowing a backing-up capability for the simulations. In the area of dynamic subsystem modeling, this paper describes the algorithms that simulate each space vehicle subsystem as well as trades between model execution time and model fidelity. Subsystems discussed include orbit determination, thermal control, electrical power, telemetry, tracking, and commanding, and attitude determination.

Overview of the Global Positioning System

GPS is a spaced-based radio navigation, time distribution and nuclear detonation (NUDET) detection system. Its mission is to provide precise, continuous, all-weather, three-dimensional (3-D) position, velocity, timing, and NUDET information to properly equipped air, land, sea, and space-based users. The system serves various military and civil operations as a highly accurate positioning and navigation data reference. Military operations include, but are not limited to, enroute navigation, low-level navigation, target acquisition, close air support, missile guidance, command and control, all-weather air drop, sensor placement, precise survey, and instrument approach. Civil applications include, but are not limited to, air, land, and sea navigation, surveying, and search and rescue. As a NUDET detection system (NDS), it detects visible light, X-rays and electromagnetic disturbances emanating from NUDETs.

GPS consists of three segments: the Space Segment; the User Segment which consists of the NUDET detection users and the navigation users; and the Control Segment which includes the Operational Control System (OCS). The Operational Control System includes the Master Control System (MCS), multiple Ground Antennas (GAs), and multiple Monitor Stations (MSs).

The GPS OCS utilizes GAs and MSs to monitor and control the GPS constellation and to receive GPS Navigation data. Ground Antennas contain equipment to receive a single GPS S-Band State-of-Health Telemetry downlink and to uplink satellite commands and Navigation Uploads to a GPS satellite. Each Monitor Station contains a GPS L-Band Antenna and a twelve channel receiver which allows it to process the Navigation Telemetry downlink from multiple GPS satellites simultaneously.

The Need for a GPS Simulator

The Air Force operations squadron that controls the GPS satellites from the MCS holds regularly scheduled training sessions for the crews that perform the operations. Until quite recently, most of these training sessions have included a large degree of 'paper training', as they had no way to adequately simulate the effects of the commands that get issued to the satellite vehicles, nor did they have any way to introduce anomalous conditions into the training system. When delivered, the GPS High Fidelity Simulator (herein referred to as the Simulator) will greatly enhance their ability to properly train crew members in day-to-day functionality and in handling anomalies that occur. It will also aid in the resolution of problems for which there is no established handling procedure. For instance, if a problem occurs on a vehicle, there may be a variety of proposed solutions for fixing it, but the effectiveness and the secondary effects of a given proposal may be unknown. High fidelity simulations will produce data to aid in selecting the best proposal. In addition, the simulator may be used for testing new MCS software releases and maintenance fixes.

GPS High Fidelity Simulator System Architecture

The intent of the Simulator is to simulate as realistically as possible the GPS system as seen by the MCS. This includes the constellation of satellites, environmental conditions, the network of GAs and MSs, and external users. Within the Simulator design, high level components are represented as *domains*, each of which addresses a common intellectual area of interest.

Figure 1 depicts the six domains of the Simulator. The five solid ovals within the Simulator boundary represent application domains. The sixth, Common Services, provides the application domains with those characteristics associated with operating systems, databases, message passing and logging.

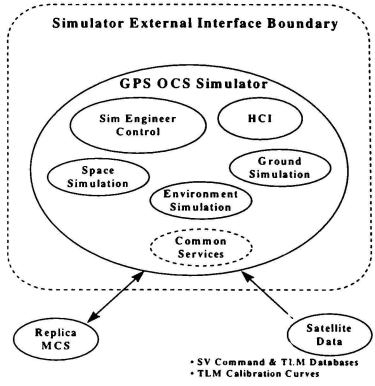


Figure 1

The Sim Engineer Control domain contains the functions which allow the simulator Engineer to control and view the simulation. This includes creating and using scripts, which are used to define both nominal and anomalous conditions. Also included are the capabilities to control the simulation speed (paused or an integer multiple of real-time) and the simulation start date/time. The Space Simulation domain simulates the subsystems physically resident on the spacecraft, as well as the position of the spacecraft in space. It also generates the simulated telemetry. The Ground Simulation domain simulates the GAs, MSs, external user agency interfaces, and the communication links between these entities. It also handles the actual communication between the simulator and the MCS. The Environment Simulation domain simulates the physical environment in which the GPS constellation operates. This includes earth polar motion, nutation, and the atmospheric effects on a transmitted signal. The HCI (Human Computer Interface) domain provides the HCI used by the simulator engineer, along with visibility of the GPS

trainee actions. The Common Services domain is the home of functions used by more than one of the other domains. Some of these are infrastructure functions such as logging and interprocess communication, but others are higher level, such as generic classes for subsystems and measurands. All domains impose requirements upon the Common Services domain.

Figure 2 maps the Simulator domains to elements of the hardware architecture. Primary considerations behind these allocations include the applications' numeric processing and input/output characteristics, the desire to reduce network traffic and operator response time, and the need to maintain sufficient processor (CPU) reserves.

The Space and Environment Simulation domains are allocated to the Simulation Engine, a Silicon Graphics (SGI) symmetric multiprocessor (SMP) containing 10 individual processors. Space Simulation, which includes simulation of the subsystems for 30 vehicles, and Environment Simulation, which includes simulation of atmospheric and space effects for those 30 vehicles, are both domains where the processing is clearly numerically intensive with relatively little demand for I/O. This characterization is especially true given the requirement for a fast forward capability of up to 25x real-time.

Another consideration that makes Space Simulation ideal for the SMP architecture is the fact that the application lends itself so well to partitioning. In particular, very little communication between spacecraft needs to be modeled by the Simulator, so each spacecraft can be statically allocated to a single processor without introducing situations where one processor must wait for another.

The other primary simulation function, Ground Simulation, is distributed across the Simulation Engine and the Simulation Communications Processor. To understand this distribution, it helps to understand a little about the architecture of the operational GAs and MSs. Equipment at the GAs and MSs includes workstations, which communicate directly, via TCP, with the MCS. The GA/MS Workstations also communicate, via TCP/IP sockets, with locally networked VME (VERSA Module Eurocard) processors at the remote site to control and status VME attached hardware devices. Hardware devices here include Servo Control Units, Telemetry Receivers, Crypto equipment, etc.

The elements of Ground Simulation allocated to the Simulation Engine are those

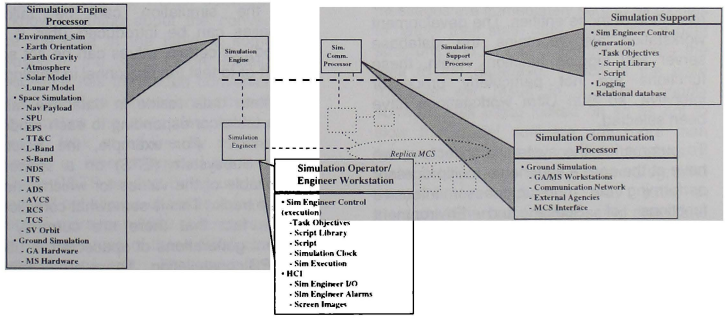


Figure 2

that model the VME attached hardware devices. These models are generally more periodic in nature than the remainder of Ground Simulation. As such, these models, along with Space Simulation and Environment Simulation, can capitalize on the use of Rate Monotonic Scheduling on the Simulation Engine.

Processing on the Simulation Communications Processor, on the other hand, is primarily event driven and more I/O intensive. A single Sun Ultra Workstation running Sun Solaris O/S was chosen as the Simulation Communications Processor to facilitate reuse of the operational GA/MS Workstation software. The Simulation Communications Processor runs multiple instances of the GA/MS Workstation software. All communications between those instances and the MCS must pass through another process which models availability of the GA/MS Workstations, network availability, and network delays. In addition, communication links to several external user agencies, which normally receive various data directly from the MCS, are also modeled on the Simulation Communications Processor.

Finally, the Simulation Engineer Control and HCI domains have been allocated to two Sun Workstations; one for development of scripts and task objectives, and another for execution of those entities. The development workstation also contains the database server and logging function. Again, these functions are not particularly processor intensive, so Sun Ultra workstations have been selected.

To summarize the system architecture, we have at the core a symmetric multiprocessor performing the more processor intensive functions of Space and Environment Simulation while a set of distributed processors implement an assortment of user and communications interfaces.

The Simulator Database

Each of the fifteen software models in the GPS simulator typically has between several hundred to a few thousand state data items that define the current state of a spacecraft subsystem or a ground component, and get updated during the course of a simulation run. A normal run will simulate multiple spacecraft and ground components, thereby multiplying the amount of state data. At initialization for a simulation, a full set of these state data items must be read into memory for each spacecraft and ground component specified for that particular run. These are termed the 'initial conditions'. At any point during a run, a set of these state data items can be saved, potentially for use as initial conditions for a subsequent run.

Operation of the simulator is controlled at the Sim Engineer Workstation by the Sim Operator/Engineer (SO/E) using the HCI domain. The three primary capabilities of the SO/E are to build scripts for training runs, preside over a simulation run (execute a previously built script), and choose or manipulate sets of state data (initial conditions) for a run. The SO/E also has displays showing the MCS software seen by any of the personnel in the room. These can be used to monitor any data points, messages or alarms, and actions taken by the trainees in the session. When preparing for a simulation, the SO/E must select a set of initial conditions from a list, after which he/she can choose to update any of the state data values prior to the initialization. In this way the simulation can be tailored, anomalies can be introduced (this is one method), or certain values can be set so as to alert or to test the personnel in training.

The state data reside in database tables, with a table corresponding to each model or subsystem. For example, the Electrical Power Subsystem (EPS) on a spacecraft has a table of the values for which it needs to keep track. This is somewhat complicated by the fact that there are currently two different generations of spacecraft in use in the GPS constellation. These are commonly referred to as block types. Although there is some overlap, each block type has a different set of state data corresponding to it. Therefore EPS really has two state data tables in the database, and the software must indicate which block type of spacecraft is in use when requesting a read or write to the database.

Since the state data values are of several different data types, this raises the question as to how best to store these values in a database table. Each column in a table must be typed, meaning that every item in that column must be of the same data type. In other words, you can't store an integer and a character string in the same column. If you think of a set of state data as simply a series of values, then it is possible to view a state data table as shown in Fig. 3 in which each row of the table contains a key for identification, followed by the data in one very wide column.

KEY	STATE DATA

Figure 3

For this approach to work however, the data in that column must all be of the same type. Therefore the GPS Simulator has chosen to convert all the state data to character type for storage in the tables, and to convert it back to the regular data types at initialization. These conversions are performed by the database interface software so that the model software need not know anything about the conversions. Since the data in the table is sometimes displayed to the SO/E before initialization, this scheme has the added convenience of not requiring data conversion for that display. It also makes the state data tables much easier to look at off-line by maintenance personnel or other interested parties. The database COTS product being used for this project will not allow a character type column to contain more than 2000 characters, so the table uses a series of these 2000 character columns in one row. The key information includes the id of the spacecraft or ground component and the name of this set of state data (or initial conditions). With this approach, one row of a table contains an entire set of state data for a particular model.

ID	SET NAME	2000 CHARS OF DATA	2000 CHARS OF DATA

Figure 4

At initialization for a simulation, requests are issued from the Software Engineer Control to read in the state data to the individual models. Typically there will be multiple spacecraft and ground components identified for the run. This means that each model will have multiple (Ada) software tasks running, with each needing to read in a set of state data corresponding to its spacecraft or ground component. After the models have been initialized and the simulation has begun, the model software will continuously update the state values in local memory. The GPS Simulator is a real-time system with a focus on time performance, therefore no direct access to the state data in the database is permitted by the model software during a run.

One of the options that the SO/E has during a simulation is to perform a 'save state' at any time. When this option is chosen, messages are immediately sent to all models to gather their state data values into a data structure and to call a routine to write the data to the database. Each of the write requests will result in a lower priority software task being kicked off so as not to slow the model software any more than necessary. This way, multiple versions of state data for any given spacecraft/ground component configuration may exist in storage. When a complete set of state data is saved, it is given a temporary name by the software. At the completion of a run, the SO/E will be prompted to view the temporarily saved sets of data and decide which ones to permanently save. When a set of state data is permanently saved it must be given a name by the SO/E. All temporary sets that are not saved are then deleted from the database. The SO/E can at that point (or any time in the future) start a new run using

any of the permanently saved sets of data as the initial conditions for the run. This scheme allows the operator the capability to back up to any chosen point in a simulation, or to simply save a configuration that may be useful for future training or testing.

The Simulator database system, in summary, allows the overall state of a simulation run to be saved at any time without unduly affecting the performance of the simulation software, and it allows user manipulation of state data for customizing runs at initialization.

Dynamic Subsystem Modeling

The dynamic subsystem modeling portion of the simulator includes the equations and algorithms needed to accurately describe the behavior of the spacecraft. Care was taken to provide an accurate dynamic simulation without sacrificing system performance. Some of the spacecraft subsystem models are discussed as well as some of the performance tradeoffs that were made. It should be noted that the constants and variables used in the equations that follow are supplied at the end of this paper.

Orbit

The orbit prediction problem has been solved many times in the past several decades. The simulator team chose to use established methods for orbit propagation. The models used include the Earth's gravity, solar gravity, lunar gravity, and solar pressure. The effects of solar and lunar gravity are computed as point mass effects. The effects of the Earth's gravity is computed based on the WGS84 model and uses zonal, tesseral, and spherical harmonics with an eighth degree and order. The effects of thruster firings are also included in the orbit propagation and will be discussed in greater detail.

Thermal Control

The thermal control subsystem computes the temperatures of various spacecraft surfaces and components. It also simulates the heaters needed to maintain a proper thermal range for various components. The thermal control subsystem is being modeled using several different techniques.

The first is a fairly high fidelity energy balance for the external surfaces and solar arrays. The governing differential equation used to compute the temperature of a given surface is given by:

$$\frac{dT_{SV}}{dt} = \frac{1}{mc}\left[\dot{Q}_{Joule-heating} + \alpha_{SV} A_{\perp} K I_{Sun} - \sigma A_{SV} \varepsilon_{SV} T_{SV}^4 + \sigma A_{SV}\left(F_{SV-Space}\varepsilon_{Space} T_{Space}^4 + F_{SV-Earth}\varepsilon_{Earth} T_{Earth}^4\right)\right]$$

For the case of solar arrays, adjustments are made to the equation to compensate for the radiation of energy from two surfaces and the absorption of solar energy from one. A separate differential equation is computed for each surface of the spacecraft. These equations are then numerically integrated to produce a new temperature state for each surface.

The second method for thermal simulation is used to compute the temperatures of the components on the interior of the spacecraft. For GPS spacecraft, many components are mounted on the internal surface of the panel that makes up the exterior structure of the spacecraft. When a particular exterior surface is illuminated, heat is transferred to the components mounted to the other side of this surface. Therefore, a simple thermal model can be constructed using the cosine of the angle of solar radiation (0° is defined as the surface normal) as a scalar to a predetermined temperature range. Separate ranges are given for operational states and non-operational states of the component. The eclipse factor, K, is also included to account for any decrease in solar radiation due to an eclipse. The equation for the modeling of internal temperatures is:

$$T = K(T_{hi} - T_{lo}) \cos(\theta) + T_{lo}$$

A full scale thermal model that divides the spacecraft into nodes and solves the heat equation at each node was considered. This method was not incorporated due to the number of nodes needed to encompass every spacecraft component. The solution to this system is computationally intensive and is not well suited for a real-time simulation.

One of the component heaters that is modeled is the battery heater. This heater is controlled by a thermostat with a high and low setpoint. The equation for the battery temperature is given by:

$$\frac{dT_{battery-N}}{dt} = \frac{1}{mC}(\eta_{heater} Q_{heater} + Q_{heat-of-reaction} - A_{battery} k_{SV} \frac{T_{battery} - T_{SV}}{L})$$

Here, it is interesting to note the dependence of the battery temperature on the temperature of the mounting surface, T_{sv}. This interaction leads to complex thermal dynamics. A plot of battery temperature and space vehicle mounting surface temperature versus time is shown in Figure 5. This plot covers several days so that the periodic effects of orbit position and spacecraft attitude can be shown. These variations due to the orbit are indicated by the solar scale. A solar scale value of one represents full sun where zero indicates that the surface is in full shade.

It can be seen in Figure 5 that the battery temperature is quickly synchronized with the exterior surface temperature. It can also be seen that the battery temperature maximum is slightly out of phase with the solar input. This is easily explained with the thermal lag due to the different heat capacities between the spacecraft mounting structure and the battery.

By making a compromise between the fidelity of the model and computational performance, an accurate simulation of the thermal behavior of the spacecraft can be produced.

Attitude

The model for the spacecraft attitude system contains the algorithms necessary to compute the angular accelerations, velocities, and positions. The attitude of the spacecraft is estimated by the use of Earth and Sun sensors. These sensors provide Earth pointing errors as well as Sun pointing errors which are then used as feed back in the attitude control loops. Once the attitude errors are detected, the attitude control system generates commands to the reaction wheels which then change speed. This change in wheel speed results in a torque applied to the spacecraft body. The attitude simulation is started with the initial attitude angles. For GPS satellites, the spacecraft +Z axis is always controlled to point to the earth and the X axis is controlled to point to the Sun. Therefore, instead of having an initial condition of three separate pointing angles, the attitude simulation uses an initial yaw angle (rotation about the Z-axis), and initial pitch and roll errors. These parameters are then propagated in the attitude simulation. Computations are made for the acceleration of the three propagated

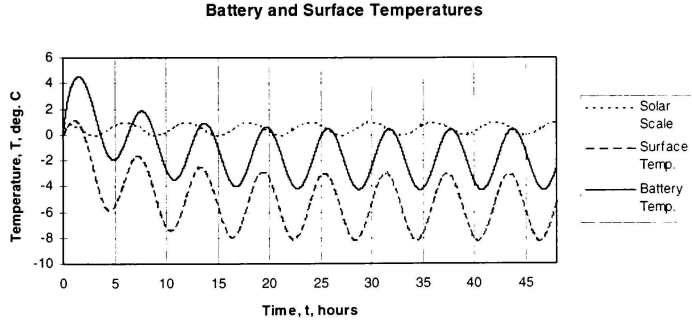


Figure 5

angles based on the torques generated by the reaction wheels and the magnetic torquers.

These accelerations are integrated to produce velocities which are then integrated to produce the new state of the yaw angle, pitch error, and roll error. From this new state, new Sun and Earth pointing vectors are computed in the spacecraft body coordinate system. These new pointing vectors are then used to compute a new yaw angle and pitch and roll errors. Again, with pitch and roll errors detected, reaction wheel speeds are changed and the process repeats. The pitch and roll error control system block diagram is shown in Figure 6 below.

Telemetry, Tracking, & Commanding

The telemetry, tracking, and commanding (TT&C) subsystem interfaces with the ground to accept commands and provide telemetry. In the simulation, this subsystem is responsible for receiving all commands and forwarding them to the appropriate subsystem. It also samples digital and analog data from other subsystems as well as receives serial data from the payload and formats this data into standard frames. This subsystem has little dynamic behavior and poses little challenge for simulation.

Electrical Power

The simulation of the electrical power subsystem involves the modeling of the batteries, solar arrays, sun sensors, and power regulation equipment. For the batteries, preset curves for the battery voltage versus battery temperature are used. Equations are then written to describe the charging behavior of the batteries based on the level of incoming current from the solar arrays. The electrical power system also contains control logic to move the solar array to track the sun. The sun sensor is mounted on the solar array and measures the solar array pitch angle error. Nominally, this sensor computes a zero pitch angle error when the solar arrays are normal to the sun in the pitch axis. This computed pitch angle error is fed back to the control logic which then commands the solar array drive motors to move. This movement then results in a new sensed pitch angle error. The block diagram showing the control loop is in Figure 7.

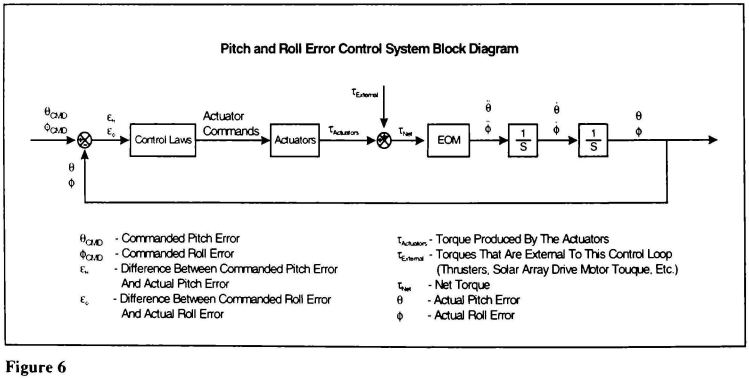


Figure 6

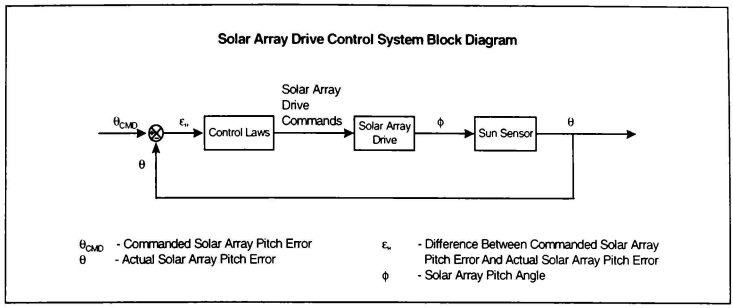


Figure 7

Reaction Control

The simulation of the reaction control system includes the spacecraft thrusters as well as the propellant tanks, and lines. To ensure that the thruster efficiency is maximum, the propellant is heated using a catalyst. The catalyst is heated using a heater that is nominally switched on prior to thruster firing. The model for the catalyst heaters is contained in the thermal control subystem.

For the thruster model, the standard rocket equation is used.[1]

$$T = \dot{m}V_e + (p_e - p_a)A_e$$

Knowing the nozzle exit area ambient pressure, and estimating the nozzle exit pressure, the solution of the thrust equation only involves the calculation of the mass flow rate and the exit velocity. Through several simplifying assumptions, such as one-dimensional, adiabatic flow, and a calorically perfect gas, an equation to model the exit velocity can be obtained.[1]

$$V_e^2 = kR_{gas}T_c \frac{1 - \left(\frac{p_e}{p_c}\right)^{\left(\frac{k-1}{k}\right)}}{k-1}$$

Making similar assumptions, an equation for the mass flow rate can be derived.[1]

$$\dot{m} = p_c A_t \left[\frac{k}{RT_c} \left(\frac{2}{k+1} \right)^{\frac{k+1}{k-1}} \right]^{\frac{1}{2}}$$

This still leaves the problem of computing values for the pressure and temperature of the combustion chamber. The temperature of the combustion chamber can be estimated knowing the performance of the catalyst heaters where the pressure is more difficult to estimate. Since the thrust performance of the spacecraft thrusters is known, simple trial and error can be used to deduce a value for the chamber pressure.

System Stability

The dynamics of the complete system is an area that deserves as much attention as the model generation. Each model has a specified execution period. This period is based on the estimated largest step size that can be accommodated while still maintaining model accuracy. Model algorithm accuracy and stability have been tested for each model, yet system accuracy and stability can not be inferred due to external factors. For example, the Rate Monotonic Scheduling algorithm attempts to execute each model at the desired period. This is a goal, not necessarily a guarantee. Under certain

extreme CPU loading conditions, model execution may not happen at the desired period. This can lead to system instability even though model stability has been proven. Therefore, when specifying execution periods, system stability must be accounted for.

Through the use of high fidelity dynamic models and lower fidelity static models, a balance can be achieved that accomplishes the goals of the real-time simulator.

Constants and Variables
Used in the Above Equations:

m Mass of the surface for which the temperature is being calculated
c Specific heat of the material for the surface
$\dot{Q}_{Joule-heating}$ The heat generated from the electrical components internal to the spacecraft
α_{SV} Absorption factor for the surface
$A_{\perp}$ The area of the surface that is perpendicular to the Sun
K Eclipse Factor
I_{sun} Solar radiation energy
A_{SV} Area of the surface that is radiating.
σ Stephan-Boltzman's constant
ε_{SV} Emmisivity of the surface
T_{SV} Temperature of the spacecraft
$F_{SV\text{-}Space}$ Amount of surface that is visible to space
ε_{Space} Emmisivity of space
T_{Space} Temperature of space
$F_{SV\text{-}Earth}$ Amount of surface that is visible to the Earth
ε_{Earth} Emmisivity of the Earth
T_{Earth} Temperature of the Earth.
T_{hi} The upper temperature limit for a particular spacecraft component
T_{lo} The lower temperature limit for a particular spacecraft component
m The mass of the battery.
C The heat capacity of the battery.
η_{heater} The efficiency of the heat transfer from the battery heater to the battery.
Q_{heater} The thermal energy transferred from the battery heater to the battery.
$Q_{heat\text{-}of\text{-}reaction}$ The thermal energy transferred to the battery due to the chemical reaction within the battery.
$A_{battery}$ Area of the battery that is in contact with the spacecraft surface.
k_{SV} The coefficient of thermal conductivity.
$T_{battery}$ The current battery temperature.
T_{SV} The current temperature of the battery's mounting surface.
L The thickness of the mounting surface.
T Thrust
$\dot{m}$ Mass Flow Rate
V_e Nozzle exit velocity
p_e Exhaust pressure at nozzle exit
p_a Ambient pressure
A_e Nozzle exit area
k Ratio of specific heats for the propellant
T_c Combustion chamber temperature
p_c Combustion chamber pressure
A_t Nozzle throat area

References

[1] Griffin M.D., and French J.R., "Space Vehicle Design", AIAA Inc., Washington D.C., 1991.

AIAA-97-3672

INTERNATIONAL SPACE STATION PART TASK TRAINERS

David J. Forrest and J. Benton Christman

Technical Consultants, GeoLogics Corporation, Houston, Texas 77058

Mark Wiederholt and Reynaldo R. Rodriguez

Hughes Training Inc., Houston, Texas 77058

Abstract

Training astronauts and ground controllers to operate and maintain the International Space Station (ISS) requires a variety of training and simulation systems. This paper describes the Part-Task Trainer (PTT) simulators that are being developed for the International Space Station. ISS PTTs will be deployed for U.S. and International Partner astronaut and ground controller training. The purpose, architecture and project history of the PTT are discussed as well as the overall ISS training approach that PTT supports.

Introduction

Training astronauts and ground controllers to operate and maintain the International Space Station (ISS) requires a variety of training and simulation systems. This paper focuses on the Part-Task Trainer (PTT) simulators that are being developed as part of the Space Station Training Facility (SSTF) at NASA's Johnson Space Center (JSC). The SSTF is being developed for the JSC Mission Operations Directorate (MOD) under the Training Systems Contract (TSC) by Hughes Training Inc. (HTI). The Simulator and Operations Technology Division of MOD is responsible for development, maintenance and operation of the system. The Space Flight Training Division of MOD is responsible for system requirements and will be the primary user of the system.

The overall training strategy for the ISS includes classroom instruction, self-study, single-purpose PC-hosted Computer-Based Training (CBT), single system training, high-fidelity simulator training for crews in the high-fidelity Full Task Trainer (FTT), and integrated simulations including the FTT and the Mission Control Center (MCC), as well as neutral buoyancy and other types of training. The PTT simulators are being developed to fulfill the single system training needs for ISS systems.

The same architecture and software models being developed for the PTT are being used as the basis for an American Segment Trainer (AST) for the Russian Space Agency (RSA). The AST will provide multi-system simulation capability and will be used in conjunction with RSA's Russian Orbital Segment (ROS) simulators to provide comprehensive ISS simulation for training of cosmonauts in Russia. This paper will describe the purpose of the PTT, how it complements the other ISS training systems, and the purpose of the AST.

The PTT is being developed in parallel with the ISS because of early training needs. The ISS hardware and software configuration will evolve as construction progresses. In addition, the experience gained and new capabilities added in each assembly step will generate new training requirements and modifications to existing training procedures. To handle these on-going changes, the PTT project is scheduled on an incremental basis. This approach providing best-available modeling of ISS systems tailored to certain configurations in the ISS assembly sequence. The overall approach and timeline of the project are discussed along with a brief history and current status.

The Purpose of the PTT

Since the ISS will eventually be permanently inhabited and continuously on orbit for at least ten years, hundreds of people - both crew members and flight controllers - must be trained to maintain and operate the vehicle. To accomplish this training task, NASA is using an approach based on the highly successful approach used for Space Shuttle training.

Students (astronauts, flight controllers, and other technical and managerial personnel) receive classroom

Copyright © 1997 by the American Institute of Aeronautics and Astronautics, Inc. All rights reserved.

training, do independent study using workbooks and Computer-Based Training, receive single system instruction in the PTT and receive integrated (typically mission specific) instruction in the FTT and/or MCC as appropriate. The Space Flight Training Division keeps records of each student's progress. Most students are required to complete at least a minimum curriculum designed for their particular assignment.

Independent study involves studying workbooks and CBT lessons on various ISS systems and procedures. Independent study occurs at levels of detail ranging from familiarization and overviews to very specific "generic" (i.e. largely mission-independent) system descriptions and procedures.

Classroom training provides supervised instruction from familiarization to fairly specific, (typically generic) knowledge to a large number of people. Classroom and workbook training are pre-requisites for moving on to simulator training.

The FTT provides the highest fidelity, full mission simulation capability. It is used almost exclusively for flight crew training. The training sessions are largely mission specific. Training for a specific mission culminates in integrated training sessions involving the FTT and a fully staffed MCC.

PTT training is used to take students from the "book learning" level of knowledge and give them "hands-on" experience with one system at a time. Students get their first real exposure to system malfunctions and recovery procedures in the PTT. Flight crews and flight controllers participate in 1-2hr sessions of one-on-one instruction at varying levels of detail. PTT training sessions provide "generic" training on single systems at a particular configuration in the Assembly Sequence. After PTT training, students are ready to move on to MCC, FTT, and integrated training.

Each tier of the training strategy fulfills certain needs. The independent study and classroom training disseminate knowledge to large numbers of people who work with ISS in some way that requires knowledge of its systems and their operation. This sort of training is inexpensive on a per person basis. The FTT, MCC, and integrated training sessions provide hands-on, detailed training for normal and contingency operations for the smaller number of people who operate the ISS directly. This sort of training is comparatively expensive on a per person basis.

PTT training provides hands-on training to more people than can be accommodated in the MCC and FTT training sessions. It also offers the opportunity to focus on the operations of one particular system without having to deal too much with the other systems. On a cost per person basis, PTT training is less expensive than FTT/MCC training but more expensive than classroom work and independent study.

Architecture

The PTT is a distributed, scalable system of Commercial Off-the-Shelf (COTS) POSIX workstations. This system runs a robust software infrastructure that controls plug-in software models, distributes data on a LAN (Local Area Network), and provides user interaction using a COTS User-Interface (UIF) development tool. For flight controller training, the PTT incorporates the workstation-based console hardware used in the MCC. For astronaut training, the PTT incorporates the ISS Portable Computer System (PCS) - a PC-compatible laptop computer. ISS display and control (hardware) panels are simulated in software via visually accurate "virtual panels".

Hardware

The basic PTT consists of four UNIX workstations: The Simulation Computer (SC) hosts the "number crunching" tasks that run the ISS system models and another task that controls the dissemination of model data and the collection of user command input. The Instructor Operating Station (IOS), as the name implies, is used by the training instructor to configure the system for a particular lesson, monitor the student's activities, and induce simulated malfunctions in the models. The Student Station (SS) is used by the student to monitor and command the simulated ISS systems. The SS is either a PCS or an MCC "Blue Console" depending on lesson objectives. The Virtual Panel (VP) is used by the student to interact with simulated hardware display and control panels.

The PTT training area at JSC consists of four of these systems for operational training plus a fifth for development and testing. Two of the four operational trainers also have an MCC "Blue Console" which contains two DEC Alpha workstations driving three monitors. This hardware is used for flight controller training. One of the other two operational trainers will be outfitted with a Soyuz control yoke for use in Soyuz crew training.

The SC is currently a Silicon Graphics, Inc. (SGI) Challenge L. This computer has 6 CPUs and 256 MB of memory. The IOS and VP workstations are SGI Indigo 2's with one CPU and 128 MB of memory each. The Student Station is currently a SGI Indigo 2 but will be replaced with actual PCS hardware in the future.

The IOS, VP, SS, and in some cases an MCC console or Soyuz control yoke are co-located in a training room. Two of the four training rooms contain MCC consoles. One of the remaining rooms contains the Soyuz equipment. The SC's are all housed in a separate room and connected to a common file server for storage and administration purposes. The DEC Alphas in the "Blue Consoles" are also connected to a common DEC Alpha file server.

Software

PTT software is divided into functional models[1] (of the ISS systems) and the "software infrastructure" which drives the models and manages LAN traffic and user interfaces. The execution of models is selectable by configuration files and software infrastructure controls. Simultaneous execution of multiple models and interaction between models is supported.

Software Infrastructure

The software infrastructure consists of a C-language "non-real-time task" (NRT), at least one Ada language "real-time" (RT) task, the SAMMI UIF tool and the PTT peer. The NRT handles LAN communications, and communications with the UIF software and external devices such as MCC consoles. All communication between the NRT and RT takes place through a shared memory segment

The PTT peer executes on the SS, IOS, and VP as needed. It handles communications between the NRT and a COTS UIF tool (SAMMI, marketed by Kinesix) by making calls to the SAMMI Application Programming Interface (API). The PTT peer communicates with the NRT over the LAN and refreshes the SAMMI displays once a second (Fig. 1).

The RT handles model sequencing and control including controls for running and freezing the simulation, capturing the simulation state and restarting from previously captured simulation states referred to as "data stores". The software infrastructure starts as many RT's as necessary. The number of RT's required is defined in configuration files and environment variables.

To take advantage of the SGI Simulation Computers' multiple CPU hardware architecture, and cope with model growth, the single CPU software infrastructure has been extended to support multiple RT's running synchronously. The current hardware allows for a maximum of 5 RT's. Although the simulation could, for some models, be run on a single CPU, the RT has to be on a different CPU than the NRT in order to guarantee real-time execution.

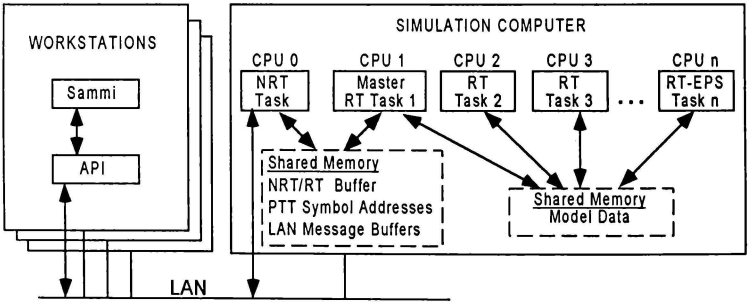


Figure 1. PTT Architecture

One of the RT's is designated the "master" RT. Any others are designated "slaves". Each RT has its own executive, contains one or more models, and is assigned to its own CPU. The frame-rate scheduling

procedure in the slave RT's executive does not have all of the functionality found in the master task's "Sequencer". The master RT creates a shared memory segment which contains all of the PTT symbols. The NRT does not interact with the PTT symbols, but simply transfers incoming and outgoing LAN messages to the RT. In order to make these LAN messages more efficient, each symbol is represented by an index into the shared memory segment and a value.

The RT's manage the execution of models across ten 100 millisecond minor frames of a 1 second major frame. The models may be configured (through configuration files) to run in 1, 2, 5, or 10 of the minor frames in each major frame (i.e. they may run at 1, 2, 5, or 10 Hz). The multiple CPU's are synchronized every minor frame. If a RT takes longer than 100 milliseconds the RT's in other CPUs will wait until it is done.

The EPS model required the introduction of a special real-time task because it needed longer than 100 msec to run. Because of its longer execution time the special EPS RT is separated from the other RTs and has a special "sequencer" in its Executive.

Models

Models of ISS hardware and software systems are developed as Ada-language software that interacts with the software infrastructure through a shared memory interface. The Guidance, Navigation and Control (GN&C)/Prop System, Electrical Power System (EPS), Thermal Control System (TCS), Environmental Control and Life Support System (ECLSS), Command and Data Handling (C&DH) System, Communications and Tracking (C&T) system, and the Common Berthing Mechanism (CBM) are modeled.

The models are implemented as Ada packages called "partitions" whose procedures are called by the software infrastructure. The partition packages contain procedures for initializing, updating, and transferring data to and from the model as well as capturing datastores and resetting the model from a datastore. For each partition, model data is passed through another Ada package, called a "Symbol_With". Each Symbol_With is mapped by the software infrastructure to a portion of the shared memory segment. The use of shared memory is sometimes criticized. In past systems it often meant that every procedure had visibility into every shared memory location so a large amount of coordination was required to keep different procedures from overwriting each other's data. The PTT Symbol_Withs effectively compartmentalize the shared memory so that the model partitions do not unintentionally overwrite each other's data. The coordination required to keep the Symbol_Withs from overlapping is handled by a pre-processor that is used to automatically arrange the Symbol_Withs in the shared memory segment.

Model Development

The decomposition of the PTT functional model software into partitions allows the development of any given model to be largely independent of the development of the other models. As a result, the developers of a given model may select a development approach that is appropriate for that model (within the context of project-level requirements and milestones).

Some developers are using Object-Oriented Design, others are using Structured Programming, and others are reusing models built for other systems. The TCS system uses a software tool called ROSE built by CAE-LINK. This tool generates FORTRAN code from PDL and FORTRAN libraries. The FORTRAN code is "wrapped" in an Ada package which adapts it to the PTT software architecture.

Developers start from existing software and new requirements submitted by the Space Flight Training Division. From there, if new software is being built, the developer may obtain copies of documentation for the system(s) being simulated and start designing and building software. In some cases, the developer will instead reuse models developed for another system - adapting it to the PTT architecture as required.

When the requirements for a particular delivery increment are implemented, the developer will first demonstrate the model informally to the Space Flight Training Division. Subsequently, formal execution of the test script will be done with Independent Verification and Testing personnel, leading up to formal acceptance by the Space Flight Training Division and the Simulator Operations and Technology Division. Any shortcomings found during formal testing are documented and fixed prior to official acceptance.

Reuse

Source code is being reused at essentially three different levels on the PTT project: "pure" reuse,

"wrappering", and "reuse with modification". Some of this reuse is planned and some is opportunistic. Planned reuse - sometimes referred to as "synthesis" - involves designing software specifically to be reused in other systems. Opportunistic reuse occurs when an existing piece of software is found to be a good candidate for use in a new application. The distinction between planned and opportunistic reuse can be fuzzy in many cases.

A set of fundamental objects such as vector/matrix manipulation and certain common enumeration types is available to all models and the infrastructure and is widely used. These packages are all taken from the FTT where they were intentionally built to be reused. This is by and large "pure" reuse. It results in a cost savings and reduction in total code count, hence development and maintenance costs, across the FTT and PTT combined.

Reuse by "wrappering" involves taking code that was developed for another system - possibly in a different language - and writing some additional code to adapt it to the local architecture (i.e. a "wrapper"). The PTT EPS model uses this approach.

The PTT EPS system uses part of a FORTRAN model built by the Boeing North American company to simulate certain EPS Orbital Replacement Units (ORU). This FORTRAN model is used to test the onboard EPS software for the ISS. The PTT EPS model uses Ada wrappers to adapt the FORTRAN model to the PTT software and architecture. Both the PTT and FTT are reusing the same FORTRAN model, with wrappers appropriate to the different simulators.

The ECLSS, GN&C and to a lesser extent other models reuse Ada source code from the FTT. The code is modified to account for differences in the infrastructures of the two simulators and extended to meet PTT-unique requirements. Since the core functionality is only coded once for two projects, a reduction in code count and maintenance cost is realized.

The PTT Project

The PTT project started in December 1990 to build PTTs for Space Station Freedom. Staffing for the project started in the spring of 1991 culminating in a System Functional Design Review in May. In September, the PTT demonstrated the viability of the chosen architecture on borrowed equipment. Following that demonstration, COTS hardware and software were procured for further development. Another demonstration was given in February 1992 using the COTS SAMMI user interface tool for the first time. At that point, development started for a January 1994 delivery to NASA for operational training. Test scripts for that delivery were delivered in May 1993 and that summer the design of Space Station Freedom began to change radically. Since the new design was still unstable but training needs were looming, it was decided to continue with the January 1994 delivery - dubbed Delivery 1.

The PTT project life cycle subsequent to Delivery 1 has been structured to deal with the fact that training must occur before and during the construction of the ISS. The construction will involve a series of several US Space Shuttle and Russian launches known as the Assembly Sequence. The Assembly Sequence leads to several different configurations of ISS hardware and software. NASA must train crews and ground-based flight controllers continuously from 18 months before the first mission is launched on through the full operational life of the ISS. The PTT must initially be able to accurately simulate a vehicle that has never flown, and for which the operating procedures are untested. As the Assembly Sequence progresses, the PTT must incorporate lessons learned in the form of changes to model behavior and changes in crew/flight controller procedures, in addition to the changes in vehicle hardware/software configuration.

PTT development is scheduled in roughly six month long increments. The requirements for each increment are defined by NASA shortly before the end of the preceding increment. In the early demonstration phase of the project, the requirements took the form of general "demo agreements" that specified increasing amounts of capability. As the project progresses toward operational capability, the "demo agreements" have been replaced by more detailed and formal instruments including test scripts and formal delivery content agreements.

The first official delivery of an ISS PTT for use by NASA (known as Delivery 1A) is scheduled for June 1997 and will be a simulation of the ISS configuration at mission 6A in the Assembly Sequence. Subsequent to that, the six month increments - known as "Training Drops" - will include updates due to training and flight experience gained, additions to simulate subsequent ISS assembly missions, and any changes due to ISS design changes or changes in training needs.

American Segment Trainer

The American Segment Trainer (AST) is being developed for NASA to be used for flight-crew training at the Gagarin Cosmonaut Training Center (GCTC) in Star City, Russia and for flight-controller training at Rocket Space Corporation (RSC)-Energia in Korolev, Russia. GCTC and RSC-Energia will build their own simulations of the Russian part of the ISS. The AST will be connected to Russian simulators to provide simulation of the U.S.-built parts of the ISS. This is the first time that NASA and Russian trainers have been connected in this manner. The AST is being developed to support two Russian configurations. The AST will connect with both the Russian Segment Training Facility (RSTF) at GCTC (Fig. 2) and the Russian Segment Mathematical Model (RSMMT) Trainer at RSC-Energia (Fig. 3). The RSTF and RSMMT include models of the Functional Cargo Block (FGB) and the Service Module (SM). A common AST interface is being built for both applications.

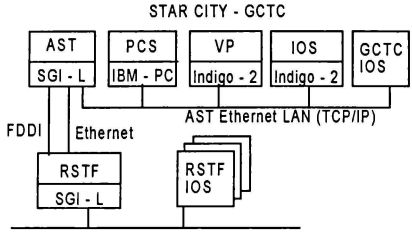


Figure 2. AST at Star City

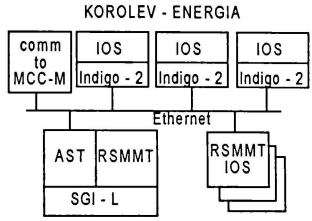


Figure 3. AST at RSC-Energia

By means of the AST interface, the PTT infrastructure can be slaved to the Russian host (RSTF or RSMMT). The Russian host will provide mode commanding, data parameters, and timing interrupts for the PTT simulation. The AST will be combined directly with the RSTF at GCTC, but the RSMMT at RSC-Energia is being designed to interact with FTT mathematical models and will interact with the AST through the Simulation Virtual Machine (SVM) architecture also developed by HTI.

All of the negotiated agreements between NASA and RSA have been placed in an Interface Control Document (ICD)[6] and described in some detail in a System Detailed Design Review (SDDR)[5]. The latter included an acceptance test (AT-1) in which a prototype of an RSTF application robustly connected and disconnected to the AST, sent mode commands to the PTT, and passed simple sets of data parameters. This test will be repeated when the AST hardware is sent to Star City.

PTT Training sessions only last a few hours and are focused on a single system. In the AST configuration, new requirements must be met. The Russian trainers are more similar to the FTT in that a training session can last 6 to 8 hours and all the models run simultaneously. The PTT's multiple CPU capability is a step toward meeting these requirements.

The RSTF and the RSMMT will both communicate with the AST through UNIX sockets. The RSTF connection is a robust single stream socket, but the RSMMT connection is a simpler connection using two datagram sockets to read and write messages. The message formats, command codes, and data parameters will be the same for both Russian hosts. The PTT timing interrupts will be replaced by data packets which will be sent by the Russian hosts at 100 millisecond intervals (i.e. 10 Hz). In the case of the GCTC, the RSTF and the AST are in two different SGI Challenge L's connected by two dedicated lines, one Ethernet and the other FDDI. Originally, the AST was to be linked to the LAN at Star City, but it was decided that having two dedicated lines made it easier to disconnect the AST from the Russian network for security reasons or testing purposes.

Agreement on which data parameters need to be passed has required a significant amount of coordination among the PTT, GCTC, and RSC-Energia developers. In some cases, there is not a one-to-one correspondence

between parameters. For example, the Russian parameters for gas composition are separate real variables, while the corresponding American parameters are in a real array. Issues such as comparable data types or the data type size also had to be considered.

Soyuz PTT and other PTT's

The flexible and robust design of the PTT infrastructure has made it appropriate for use with other trainers associated with the International Space Station. A trainer for the Soyuz crew transfer vehicle is already in development, and trainers for the National Space Development Agency of Japan (NASDA) and European Space Agency (ESA) ISS segments are being planned.

The Soyuz crew transfer vehicle will be used as an emergency crew rescue vehicle, therefore it is necessary for cosmonauts and astronauts to be familiar with its features. The Soyuz PTT is being built to provide Soyuz training capability in the U.S. The Soyuz PTT will use the PTT infrastructure together with Russian models of the Soyuz vehicle. After only a few weeks of training on the PTT Executive, Russian programmers at RSC-Energia have already begun to develop and integrate their Soyuz models with the PTT infrastructure. Unlike the PTT models which are developed in Ada, the Soyuz models will be developed using C. The Soyuz PTT will also incorporate some specialized hardware -- a Hand Controller, that serves as a third-level back-up system. The Hand Controller will be connected to the Soyuz PTT through a VME board. The Soyuz PTT will use both SAMMI displays and a Russian 3-D GUI interface.

Summary

The training of astronauts and flight controllers for construction, operation, and maintenance of the International Space Station will use a multi-tiered approach similar to that used for Space Shuttle training. The ISS Part Task Trainer will fulfill the requirements for single system, moderate cost training for astronauts and flight controllers training at Johnson Space Center. Russia will use the PTT as an American Segment Trainer in conjunction with its simulators. Other International Partners may use PTT-based simulators as well.

The PTT architecture is robust. It makes extensive use of COTS hardware and software for flexibility. This combination of robustness and flexibility makes the ISS PTT a good candidate for various forms of training involving diverse systems.

References

[1] Fishwick, Paul A., *Simulation Model Design and Execution: Building Digital Worlds*, Prentice Hall, Englewood Cliffs, New Jersey, 1995

[2] American Segment Trainer Requirements, Contract NAS9-18181, JSC-36269, DRLI-25a, Hughes Training, Inc. Houston Operations, February 28, 1995.

[3] Space Station Manned Base to Russian Segment Software Interface Control Document, SSP 50097, February 8, 1995

[4] U.S. On-Orbit Segment Pressurized Mating Adapter-1 to Russian Segment FGB Interface Control Document, SSP 42121, June 30, 1995.

[5] American Segment Trainer System Detailed Design Review, NAS9-18181, DRLI-27, Hughes Training, Inc. Houston Operations, October 9, 1996

[6] Space Station Training Facility and American Segment Trainer (AST) To Russian Segment Trainer (RST) And RSA Russian Segment Simulation Interface Control Document, SSP 50069, DRAFT, December 14, 1996.

IMPROVED METHOD OF GENERATING CONTROL SYSTEM MODEL USING MODAL SYNTHESIS

by

Tarun Ghosh
Boeing North American, Inc.
Canoga Park, California

ABSTRACT

Component mode synthesis method of analyzing large structures is a very powerful and efficient tool available in many widely used finite element codes. The method is mostly used in structural analysis. This paper shows how modal synthesis can be used to generate control system models. This paper presents a method in which starting with very detailed mathematical models of components with thousands of degrees of freedom, a structural dynamics model can be developed and eventually converted to a controls model of only a hundred or so degrees of freedom. Degree of freedom reduction is performed at several levels based on different criteria. The advantage of this method is that more accurate controls models can be generated in much less time. The method has been successfully used in the International Space Station project which is a joint international effort of many countries. The theory and method involved is general and useful for other structures using other finite element codes.

INTRODUCTION

In the present day, large structures often require the combined effort of several organizations. The International Space Station is one such structure. It is being built as an international venture and countries across the globe are participating. It is a giant station which will operate in space and mankind will use it for research, commercial production and voyage to outer space. When completed, it will be about 300 feet long.

As different organizations build parts of the station, they also generate finite element models of these parts. The models must be assembled to develop system model that could be used in loads analysis and control-structure interaction study. The loads in on-orbit are primarily dynamic in nature. All this makes component mode synthesis method an indispensable tool in the analysis of space station.

Component mode synthesis method is a technique in finite element analysis. The technique involves representing a system as an assemblage of components and is described in Craig[1], Craig[2] and Cook, et al[3]. In the system model each component is represented in terms of its boundary and modal (generalized) degrees of freedom. Any loads applied at the interior of components need to be transferred to the boundary and generalized degrees of freedom. Similarly, the response obtained at the system level is that of the retained degrees of freedom in the system model. By using the component attachment and vibration modes, the response at the interior of components can be determined. The purpose of this paper is to show how mathematical models for control-structure interaction study can be efficiently generated from mathematical models used in loads analysis.

The International Space Station is a flexible structure. In on-orbit analysis, the participating modes are below 10.0 Hz. The solar arrays which will have solar cells to convert solar energy to electricity, are very flexible. As they face the sun to absorb energy, they could be disoriented due to station loads. Therefore, there is a need to develop a controls system model to design a controller. The KU-band Antenna that will transmit radio signals to earth, will also be disoriented unless there is a controller. To develop a controls system model from scratch is a fairly time consuming task.

One method of saving time is to use the structural loads model. But the structural loads model has too many degrees of freedom and is not suited for controls study. So one has to use techniques of degree of freedom reduction to come up with a limited set of equations. The new model for control-structure

Copyright © 1997 by Boeing North American, Inc.
Published by the American Institute of Aeronautics and Astronautics, Inc. with permission.

interaction study will have the static and dynamic characteristics of the detailed model.

One efficient method of reducing degrees of freedom is modal synthesis. In this method, one generates the system model not from finite element models, but from component mode models. The component mode models have fewer degrees of freedom than detailed finite element models. Yet, the component mode models have the same static and dynamic properties as the detailed finite element models. Then the system model will also have fewer degrees of freedom. For the flexible solar arrays, there will be simply too many modal degrees of freedom. Based on strain energy, the dominant modes could be selected. This would considerably reduce the size of the system model. For controls study, not all of the remaining modes are important. So based on the controller measurements and the approximate balanced singular value method, only the dominant modes could be selected.

Problem arises when one has controller inputs and outputs at interior degrees of freedom of components. For example, the tip deflection of a solar array mast is a measurement item. In the modal representation of components, interior degrees of freedom do not appear. So the system level equations will not have the interior degrees of freedom. One method of getting around this problem is to treat interior degrees of freedom as unrestrained boundary degrees of freedom as is done in Masters and Crawley[4]. But this would make the component mode model in controls study different from the one used in loads study and has to be recreated. Also, this would increase the size of the matrices representing the component model.

An efficient method of getting around the problem is to obtain the matrix relating the response (displacement) at the interior degrees of freedom to the physical (boundary) and generalized (modal) degrees of freedom of this component. This matrix is called the (displacement) data recovery matrix. Then from the modal displacements of the boundary and generalized degrees of freedom of the component, one can perform a matrix multiplication and obtain the modal response (displacement) at the interior degrees of freedom of the component.

It can be shown that for transferring interior loads to the boundary and generalized degrees of freedom the load transformation matrix can be used which is a transform of the displacement data recovery matrix.

The work done here is an extension of (and based on) an improved method of implementation of modal synthesis in loads analysis by Ghosh[5].

METHODOLOGY

In finite element analysis, a physical model of a structure with applied loads is represented mathematically as

$$[m]\{\ddot{u}\}+[c]\{\dot{u}\}+[k]\{u\}=\{f(t)\} \tag{1}$$

$[m]$, $[c]$ and $[k]$ are the mass, stiffness and damping matrices respectively. $\{u\}$, $\{\dot{u}\}$ and $\{\ddot{u}\}$ represent the generalized displacements, velocities and accelerations at the physical degrees of freedom and $\{f(t)\}$ represents the generalized forces at the physical degrees of freedom. The equations are generated by assembling elements defined mathematically in the physical domain.

For linear elastic structures, the equations in (1) can be transformed to the modal domain and expressed as follows

$$[\mathrm{M}]\{\ddot{\eta}\}+[\mathrm{C}]\{\dot{\eta}\}+[\mathrm{K}]\{\eta\}=\{\mathrm{F}(t)\} \tag{2}$$

where $[\mathrm{M}]$, $[\mathrm{C}]$ and $[\mathrm{K}]$ are the system modal mass, stiffness and damping matrices. $\{\eta\}$ is the generalized (modal) displacement vector and $\{\mathrm{F}(t)\}$ is the modal load vector. The method of generating equations in (2) from equations in (1) requires an eigensolution of the undamped free system. That is, in equations (1), $[c]$ is assumed to be a null matrix and $\{f(t)\}$ is assumed to be a null vector. The eigenvectors so obtained are used to generate an eigenvector matrix $[\Phi]$. For linear systems,

$$\begin{aligned}\{u\}&=[\Phi]\{\eta\}\\ \{\dot{u}\}&=[\Phi]\{\dot{\eta}\}\\ \{\ddot{u}\}&=[\Phi]\{\ddot{\eta}\}\end{aligned} \tag{3}$$

Then, by replacing $\{u\}$, $\{\dot{u}\}$ and $\{\ddot{u}\}$ in (1) by the right hand sides of (3) and pre-multiplying both sides

by $[\Phi]^T$, one obtains (2). Actually, [C] in (2) comes from test data or other source. For details of the method, one can refer to Cook[5] and Meirovitch[6].

The method above is the standard method of solving structural problems and is implemented in many finite element codes. The modal synthesis method is a slight variation of this method. It is also referred to as dynamic superelement or substructure analysis. Over the years, the method has proliferated and there are several forms of it. Some of these are described in Craig[1], Craig[2] and Cook, et al[3]. An improved method of generating a control system mathematical model using modal synthesis will be shown.

The equations in (1) can also be generated by assembling component mode models. In the standard method, a system model was treated as an assemblage of finite elements. In modal synthesis, a system model is treated as an assemblage of components or substructures. The motion (of the interior) of a component can be represented either in the form of equations (1) or equations (2). The motion at the degrees of freedom that are at the boundary of a component is always represented in the form of equations (1). Since a component has a boundary and an interior, a component mode model has both physical (boundary) and generalized (modal) degrees of freedom. It consists of

$[M_C]$ = component mass matrix

$[K_C]$ = component stiffness matrix

$[T_C]$ = component data recovery matrix

$[L_C]$ = component load transformation matrix

The details for obtaining the above matrices follows. Simply stated, it is assumed that the motion of the interior degrees of freedom of a component can be obtained from the summation of two types of motion as follows:

$$\{u_{CI}\} = [\Phi_{CT}]\{u_{CB}\} + [\Phi_{CQ}]\{u_{CQ}\} \tag{4}$$

where,

$[\Phi_{CT}]$ = Component attachment modes

$[\Phi_{CQ}]$ = Component vibration modes

$\{u_{CB}\}$ = Component boundary displacements

$\{u_{CI}\}$ = Component interior displacements

$\{u_{CQ}\}$ = Component modal displacements

In (4) the part corresponding to $[\Phi_{CT}]$ is from the static equilibrium of the component when there is motion of the boundary. The part corresponding to $[\Phi_{CQ}]$ is from the dynamic equilibrium of the component when the boundary is fixed.

Let,

$$[\Phi_C] = \begin{bmatrix} [I] & [0] \\ [\Phi_{CT}] & [\Phi_{CQ}] \end{bmatrix} \tag{5}$$

Then,

$$\begin{Bmatrix} \{u_{CB}\} \\ \{u_{CI}\} \end{Bmatrix} = [\Phi_C] \begin{Bmatrix} \{u_{CB}\} \\ \{u_{CQ}\} \end{Bmatrix} \tag{6}$$

The equations of motion for the component (neglecting damping at the component level) can be represented as

$$\begin{bmatrix} [m_{CBB}] & [m_{CBI}] \\ [m_{CIB}] & [m_{CII}] \end{bmatrix} \begin{Bmatrix} \{\ddot{u}_{CB}\} \\ \{\ddot{u}_{CI}\} \end{Bmatrix} + \begin{bmatrix} [k_{CBB}] & [k_{CBI}] \\ [k_{CIB}] & [k_{CII}] \end{bmatrix} \begin{Bmatrix} \{u_{CB}\} \\ \{u_{CI}\} \end{Bmatrix} = \begin{Bmatrix} \{f_{CB}\} \\ \{f_{CI}\} \end{Bmatrix} \tag{7}$$

The form of (7) is similar to that of (1), except that the matrices and vectors of (7) are partitioned. The subscripts B, I and C represent boundary, interior and component, respectively. Therefore, $[m_{CBI}]$ represents boundary (B) mass of component (C), excited by motion of interior (I) degrees of freedom. Differentiating both sides of equations (6) with respect to time gives

$$\begin{Bmatrix} \{\dot{u}_{CB}\} \\ \{\dot{u}_{CI}\} \end{Bmatrix} = [\Phi_C] \begin{Bmatrix} \{\dot{u}_{CB}\} \\ \{\dot{u}_{CQ}\} \end{Bmatrix}$$

$$\begin{Bmatrix} \{\ddot{u}_{CB}\} \\ \{\ddot{u}_{CI}\} \end{Bmatrix} = [\Phi_C] \begin{Bmatrix} \{\ddot{u}_{CB}\} \\ \{\ddot{u}_{CQ}\} \end{Bmatrix} \tag{8}$$

Substituting equations (6) and (8) in (7) and pre-multiplying both sides of (7) by $[\Phi_C]^T$ gives

$$[M_C] \begin{Bmatrix} \{\ddot{u}_{CB}\} \\ \{\ddot{u}_{CQ}\} \end{Bmatrix} + [K_C] \begin{Bmatrix} \{u_{CB}\} \\ \{u_{CQ}\} \end{Bmatrix} = [L_C] \begin{Bmatrix} \{f_{CB}\} \\ \{f_{CI}\} \end{Bmatrix} \tag{9}$$

where

$$[M_C] = [\Phi_C]^T \begin{bmatrix} [m_{CBB}] & [m_{CBI}] \\ [m_{CIB}] & [m_{CII}] \end{bmatrix} [\Phi_C]$$

$$[K_C] = [\Phi_C]^T \begin{bmatrix} [k_{CBB}] & [k_{CBI}] \\ [k_{CIB}] & [k_{CII}] \end{bmatrix} [\Phi_C] \qquad (10)$$

$$[L_C] = [\Phi_C]^T$$

For downstream processing (assembling system model) one needs the component mass, stiffness and load vectors. For upstream processing (component data recovery), one needs the data recovery matrix. Here, the word downstream means a later step and the word upstream means an earlier step. The system model can then be assembled from the component models and would be as follows

$$[M]\{\ddot{h}\} + [C]\{\dot{h}\} + [K]\{h\} = \{F(t)\} \qquad (11)$$

where

$$[M] = \sum [M_C]$$

$$[K] = \sum [K_C] \qquad (12)$$

$$\{F(t)\} = \sum \{F_C(t)\}$$

The system damping matrix is not needed when integrating the system model from the component models. It is assumed to be proportional to mass and stiffness matrices as in Meirovitch[4]:

$$[C] = \alpha[M] + \beta[K] \qquad (13)$$

Note that while the system model in the standard method as shown by the equations in (1) has only physical degrees of freedom, the system model in the modal synthesis method as shown by the equations in (11) has both modal and physical degrees of freedom. So (11) is the modal synthesis counterpart of (1) of the standard method and the rest of the analysis is as in the standard method. As in the standard method, the equations in (11) can be transformed to modal domain and expressed as in (2) which is a set of second-order uncoupled differential equations. The mass, stiffness and damping matrices are, therefore, diagonal.

In modal synthesis, the interior degrees of freedom of a component are not carried upstream to the system level. Accordingly, no loads can be applied to component interior degrees of freedom at the system level. At the system level (equations (9)) only the component boundary and generalized degrees of freedom are available. These are referred to as analysis (retained) set degrees of freedom. So one could use component load transformation matrix, $[L_C]$, to express interior loads in terms of analysis set degrees of freedom. Similarly, to compute responses at interior degrees of freedom from the analysis set degrees of freedom responses, one would use the component (displacement) mode data recovery matrix, $[T_C]$. The elements of $[T_C]$ and $[L_C]$ are obtained from component attachment and vibration modeshapes.

It is not necessary (and not a good practice) that the load transformation matrix transfer component boundary loads. The boundary grid is often part of more than one component and appears in the system model. Therefore, component boundary loads may be applied directly to the boundary (at the system level) and at the component level $\{f_{CB}\} = \{0\}$. The equivalent boundary and modal loads of the component interior loads would be

$$\begin{Bmatrix} \{f_{CB}\} \\ \{f_{CQ}\} \end{Bmatrix} = \begin{bmatrix} [\Phi_{CT}]^T \\ [\Phi_{CQ}]^T \end{bmatrix} \{f_{CI}\} \qquad (14)$$

Then the load transformation matrix in (10) can be replaced by

$$[L_C] = \begin{bmatrix} [\Phi_{CT}]^T \\ [\Phi_{CQ}]^T \end{bmatrix} = \begin{bmatrix} [\Phi_{CT}] & [\Phi_{CQ}] \end{bmatrix}^T \qquad (15)$$

It is clear from equations (15) that for any interior degree of freedom with applied load, one needs to know the modal displacements at that degree of freedom in the attachment and vibration modes. So the contents of $[L_C]$ would be the modal displacements at interior degrees of freedom under unit positive load.

The system level force vector in (12) will be replaced by

$$\{F(t)\} = \sum \{F_C(t)\} + \sum \{f_{CB}(t)\} \qquad (16)$$

The first summation on the right hand side of (16) corresponds to forces at interior degrees of freedom of component which do not appear in the system model. The second summation corresponds to degrees of

freedom (such as component boundary) which appear in the system model. In principle the task is simple. But in practice, it is somewhat complicated by the bookkeeping requirement. The process could be automated by using some simple computer programs.

Note that a component mode model has both physical (boundary) and generalized (modal) degrees of freedom. For downstream processing (assembling system model) one needs the component mass, stiffness and load transformation matrices. For upstream processing, one needs the data recovery matrix. Note that the controls-structure interaction study model generally has a fewer number of degrees of freedom with inputs and outputs than in a loads model. Therefore, to simplify the application of loads and recovering response, it is convenient to define the degrees of freedom with inputs and outputs as boundary. Alternatively, the system modeshapes can be modified to include the modal displacements at selected interior degrees of freedom without regenerating the component mode models.

An important thing to note is that modes are normalized with respect to maximum modal displacements (not mass) in the component mode models. Also, it should be noted that not all vibration modes are used in equations (5). This makes modal synthesis method approximate. But the accuracy from modal synthesis is comparable to that from non-modal synthesis approach. This is because, while in non-modal synthesis method truncation of modes is done at the system level only, in modal synthesis, truncation is performed at both component and system level. It is like distributing the error at several levels rather than at one level.

In order to perform the control-structure interaction study, the mass, stiffness and damping matrices in modal coordinates of the system model are required. The modal displacements at the input and output degrees of freedom are required to generate the modal load vectors and obtain the modal responses.

To represent the behavior of the system in physical coordinates and apply loads the system model must be represented in physical coordinates. So the transformation matrix that relates the physical degrees of freedom of interest to the modal coordinates is also required. The elements of this matrix are simply the modal displacements. The rows of this matrix correspond to the physical degrees of freedom of interest and are the points of the system where there will be inputs and outputs. The columns of the transformation matrix correspond to the modal degrees of freedom.

While in loads analysis it is customary to have thousands of modal degrees of freedom, the models used in controls-structure interaction study typically have less than 100 degrees of freedom. There are several algorithms for selecting dominant modes. Some algorithms retain modes based on dominant frequencies of excitation. Some algorithms are based on effective weights of modes. Some are again based on retaining modes that have large modal response if all modes participated equally. The method used to select the dominant modes in this study is the approximate balanced singular value method. Note that Masters and Crawley[4] also used the same method.

Finite element codes used in loads analysis, often allow behavior of degrees of freedom of interest to be expressed in different coordinate systems. It is much simpler to write the controls relationships if all inputs and outputs are expressed in the same coordinate system (i.e. the basic coordinate system). This is achieved through a simple matrix multiplication.

The data recovery matrix based on the system model will have only the physical degrees of freedom that were defined as boundary degrees of freedom in the component mode models. For example, in the solar array component mode model only the base point (mast-canister) was defined as boundary. The degrees of freedom associated with the tip of the solar array were interior degrees of freedom of the solar array component mode model. Accordingly, the system model in physical degrees of freedom will not have the mast tip behavior. Instead of recreating the component mode models where all degrees of freedom of interest are defined as boundary as was done in Masters and Crawley[4], a much simpler method is to generate the 'modified system modeshapes' from the 'original system modeshapes' as follows.

First, the superelement boundary generalized displacements are transformed to the displacement coordinate system of the boundary degrees of freedom as follows

$$\begin{Bmatrix} h_{CB} \\ h_{CQ} \end{Bmatrix}_C = [T_0] \begin{Bmatrix} h_{CB} \\ h_{CQ} \end{Bmatrix}_0 \qquad (17)$$

where $[T_0]$ is a coordinate transformation matrix. Then , from (4),

$$\{h_{CI}\} = [\Phi_{CI}]\{h_{CR}\} + [\Phi_{CQ}]\{h_{CQ}\} \quad (18)$$

The component interior displacements computed in corresponding coordinate system of the degrees of freedom are transformed to system basic coordinate system as follows

$$\{h_{CI}\}_0 = [T_C]\{h_{CI}\}_C \quad (19)$$

$[T_C]$ is a coordinate transformation matrix. The component interior displacements in system basic coordinate system are then added at the end of the 'original system modeshapes' to obtain the 'modified system modeshapes' as follows

$$\{h_S\}_m = \begin{bmatrix} I & 0 \\ 0 & I \end{bmatrix} \begin{Bmatrix} \{h_S\}_r \\ \{h_{CI}\}_0 \end{Bmatrix} \quad (20)$$

$\{h_S\}_r$ and $\{h_S\}_m$ are the original and modified system modeshapes, respectively. Note that the operation of equation (20) must be performed for each system modeshape.

The equations of motion of the system model in modal coordinates will be

$$[M_S]\{\ddot{h}\} + [C_S]\{\dot{h}\} + [K_S]\{h\} = [T_S]\{F(t)\} \quad (21)$$

where $[M_S]$, $[C_S]$ and $[K_S]$ are the mass, damping and stiffness matrices in modal coordinates. $[T_S]$ is the transformation matrix. Each column of $[T_S]$ contains the modal displacements at input and output degrees of freedom for a system mode. Then the response in the form of displacements, velocities and accelerations at the output degrees of freedom can be easily computed from $\{h\}$, $\{\dot{h}\}$, $\{\ddot{h}\}$ and $[T_S]$ as in equations (3).

CASE STUDY

The usefulness of the new method in solving problems will now be demonstrated through an actual case study. The case is that of International Space Station (ISS) which is a joint international effort. Figure 1 shows Configuration 11 (Flight 6A) of the ISS. Of interest are the equations of motion (21) of the system in modal coordinates. Clearly, the mass, stiffness and damping matrices are diagonal. The mass matrix is an identity matrix. The terms of the stiffness matrix are simply ω_n^2, where ω_n is the natural frequency in radians. The terms of damping matrix are $2\zeta i\omega_n$ where ζ is the viscous modal damping. Typically, viscous modal damping is assumed to be about 1.0%. So, to set up the controls equations of motion one needs to know only the eigenvalues (frequencies) and the modal displacements at input/output degrees of freedom for each system mode.

MSC/NASTRAN[7], a widely used finite element code in industry is used in the structural loads analysis in the International Space Station project. MSC/NASTRAN was used to generate the component mode and system mode models. The solar array finite element model has about 6000 degrees of freedom. The component mode model has 842 modes below 7.0 Hz. Based on strain energy of modes, only 61 vibration modes were found to be dominant and were retained. The system model had 300 vibration modes. Based on approximate balanced singular value method of COSTIN[8], the dominant modes in controls study was reduced to about 100.

Though all inputs are at the analysis set degrees of freedom of the system model, some outputs were at the interior of components. The solar array mast tip displacement was an important parameter in the controller design So was the tip deflection of the KU-Band antenna. So, equations (18), (19) and (20) were used to generate the 'modified system modeshapes' that included the solar array tip displacements. The KU-band Antenna tip was made part of $\{h_S\}_r$.

The flowchart of the process is shown in Figure 2. FORTRAN programs SEDISP and PRE-COSTIN are processors used to automate the task.

DISCUSSION

Control-structure interaction study has two parts. In the first part, the dynamic characteristics of the system are obtained. In the second part, the controls equations are set up where the dynamic characteristics are used and the interaction study is performed. Loads are applied and responses are computed in physical coordinates in the second part.

There are a few things to note about the structural model. It is assumed that the system behavior is linear and elastic. There is no feedback. Besides the six rigid body modes, the system will have mechanism modes. The mechanism motions arise from allowing relative motion at gimbals. The two beta gimbals and the two KU-Band antenna gimbals will give four mechanism modes.

This paper shows how the mass, stiffness, damping and modeshape matrices of the structure can be efficiently generated. These matrices will be used to generate the complete controls model which will have feedback and motor characteristics.

Component mode synthesis is a very time consuming and elaborate effort. When that effort has already been spent to develop a system model for loads analysis, it is wise to make use of it for control system model generation. In the case study above, the same component and system models as used in the structural dynamics analysis were used in the controls study. The original modeshapes did not have the displacements at certain locations of interest. So a scheme was developed to modify the modeshapes without regenerating any of the component or system model.

The system model from loads analysis had used modal reduction at several levels. Higher order modes of the component and system model were eliminated. The solar array modal reduction was based on strain energy of modes in critical locations. An extra level of modal reduction was performed for the control-structure interaction study. The approximate balanced singular value method was used. Note that the method is based on modal displacements at controller input and output locations. Other methods based on modal mass, strain energy or frequencies of excitation could have been used.

ACKNOWLEDGMENT

This work was performed at Rocketdyne Division, Boeing North American, Inc. The material presented here is based on work performed for the International Space Station under NASA contract HX3210. The author wishes to acknowledge the support of Rocketdyne and NASA management.

REFERENCES

1. Craig, R. R., "A Review of Time-Domain and Frequency Domain Component Mode Synthesis Methods," Int. J. Analytical and Experimental Modal Analysis, Vol. 2, No. 2, 1987.

2. Craig, R. R., "Structural Dynamics", John Wiley & Sons, New York, 1981.

3. Cook, R. D., Malkus, D. S. and Plesha, M. E.,"Concepts and Applications of Finite Element Analysis", John Wiley & Sons, 3rd Edition, 1989.

4. Masters, B. P. and Crawley, E. F.,"Evolutionary Design of Controlled Structures Systems", 38th AIAA/ASME/ASCE/AHS/ASC Structural Dynamics, and Materials Conference, Proceedings, Kissimmee, Florida, April 7-10, 1997.

5. Ghosh, T. K. , "Improved Method of Modal Synthesis in the analysis of International Space-station Structures", 38th AIAA/ASME/ASCE/AHS/ASC Structural Dynamics, and Materials Conference, Proceedings, Kissimmee, Florida, April 7-10, 1997.

6. Meirovitch, L., "Elements of Vibration Analysis", Second Edition, McGraw-Hill Book Co, NewYork, 1986.

7. MSC/NASTRAN Quick reference Guide, Version 68, The MacNeal Schwendler Corp., Los Angeles, California, August 1996.

8. CO-ST-IN User's Manual, SDRC, Inc., San Diego, California 92130, SDRC Project No. 40922-4, October 3, 1989.

FIGURE-1: INTERNATIONAL SPACE STATION CONFIGURATION 11

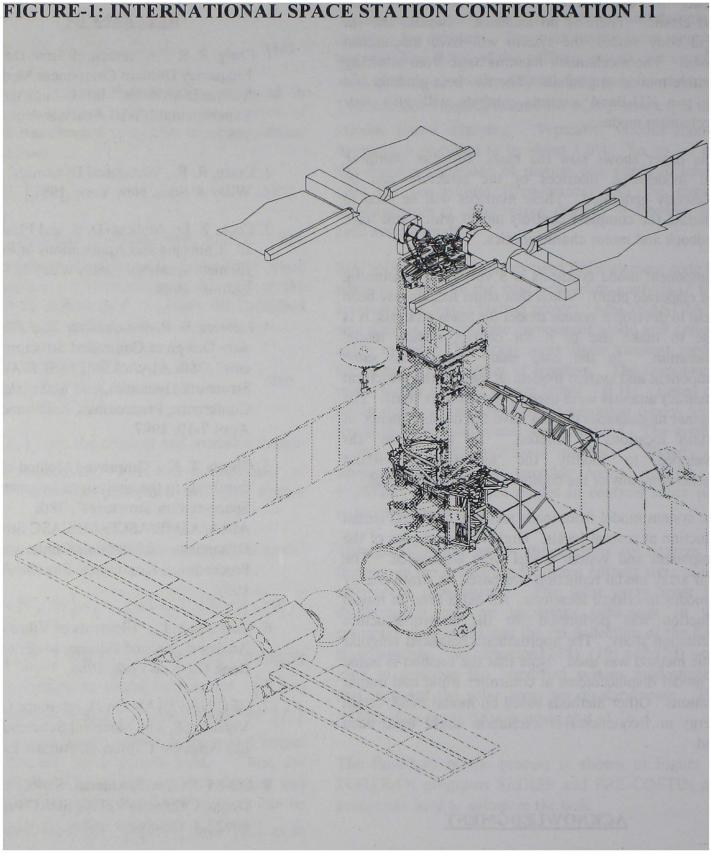
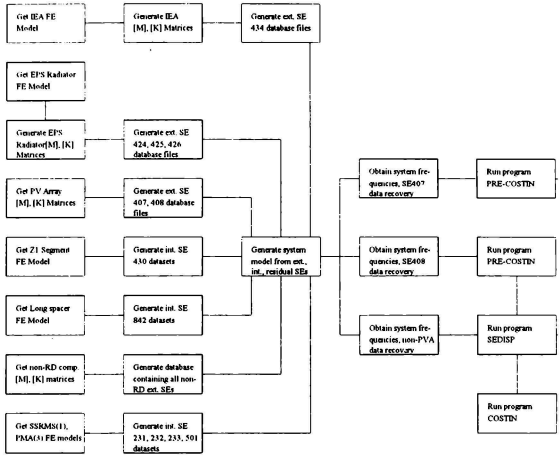


FIGURE-2: PROCESS FLOWCHART



PC-BASED DISTRIBUTED PARALLEL PROCESSING FOR REAL-TIME SIMULATION OF SPACECRAFT ATTITUDE CONTROL

Yiing-Yuh Lin* and Sern-Sih Liu†
Department and Institute of Aeronautics and Astronautics
National Cheng Kung University
Tainan, Taiwan, R.O.C.
E-Mail: fc16yyln@mail.iaa.ncku.edu.tw

Abstract

This paper presents the study of a personal computer (PC) based platform for the real-time simulation of a dynamic system. The platform consists of three parts: the digital controller module, the dynamics/environment module, and the AD/DA interface module. First two modules are the software simulation programs and are hosted by two separate 486 PCs. The third module includes four interface cards and respective driving software, where two of the cards are for digital-to-analog data conversion and the other two cards are for analog-to-digital data conversion. Connecting the two 486 PCs with the interface module allows the platform to closely emulate the hardware/software interactions between the controller and the dynamics of an actual system. Under the proposed configuration, the platform performs the simulation in a distributed parallel processing fashion as the controller module and the dynamics/environment module are running in respective PC with different speed. A low earth orbiting rigid satellite under the effects of several environmental disturbances was implemented. Three attitude control laws for the vehicle had been tested and the results were compared with the simulations by a none real-time PC.

1. Introduction

For most reported studies, the closed-loop control simulations were performed non real-time on a personal computer, a workstation, or a mainframe with sequential process, in which the control efforts and the system dynamic responses are computed by turns with a given time-step. This kind of digital simulation neglects several effects in an actual system, such as digital-to-analog and analog-to-digital signal conversions, time constants of different subsystems, and different data transmission rates by different subsystems, which may be vital to the performance of the designed controller. Especially, the frequency of onboard processor is slower than the sampling frequencies of many sensors, and a fixed time-step sequential process may not properly represent the real control case. A computer dedicated for real-time simulation is quite expensive and the simulation will be all digital under clock controlled integration step. Some question about the effects of signal conversions from digital-to-analog or analog-to-digital will still need to be examined before any prototype is built.

Recent advances in the technologies of the personal computer and its interface equipment have made possible to build a computationally efficient and inexpensive real-time simulation system[1,2]. Instead of using a high priced real-time computer, this research establishes a real-time simulation platform by using two personal computers and four data conversion interface cards to perform real-time control simulation of dynamic systems. The platform consists of two PCs, where one PC is used as the controller and the other PC emulates the dynamics and environment of a vehicle. The simulation is executed by running the controller module and the dynamics/environment module in parallel with proper time-step adjustments of each module.

An example case of real-time attitude simulation of a rigid spacecraft is included in this paper to demonstrate the application of the platform and to compare the simulation results of three control designs with the results from single PC. An optimum filter is also included in the controller to estimate attitude angles from the angular rates, which are considered as the masurements from a onboard 3-axis rate gyros. The attitude angles com-

*Associate Professor
†Graduate Student

Copyright © 1997 by the American Institute of Aeronautics and Astronautics, Inc. All rights reserved.

puted and sent out from the dynamic module are taken as the masurements from the low bandwidth optical sensors and are used to update the angle estimation of the filter.

2. Simulation Platform Setup

Two 486 personal computers are used to form the simulation platform as shown in figure 1. The controller PC-486(DX-33) contains the controller/filter software which estimates system states and computes feedback control commands. It takes in the measurements from the system dynamics and the environment conditions through analog to digital (A/D) interface card and sends out control commands through digital to analog (D/A) interface card. The system PC-486(DX2-66) receives the control signals from the controller through A/D interface card and integrates the system dynamics model and the environment model. At the end of each integration step, the system states and the environment conditions are sent out to the controller PC through D/A interface card.

The two A/D cards are the Advantech® 818 with 8 A/D differential channels, where each channel has 12-bit resolution and ±10V range. Single channel data rate for 818 card is 1 kHz but the data rate for running 8 channels will be down to 950 Hz due to multiplexing of the channels. The two D/A cards are the Advantech® 726 with 6 D/A channels of 12-bit resolution and ±10V range. The data rate is about the same as the 818 card. To make the simulation as real as possible, the integration time-step of each PC has to be adjusted similar to the data transmission frequencies of the controller and the system sensors. Output D/A from the digital controller module has slower data transmission rate than the output D/A from the dynamics/environment module. Turbo C is used for software program coding.

3. Control and Filter Design

Consider the attitude motions of the spacecraft. In the derivations of the mathematical model of a rigid spacecraft, the attitude equations can be separated from the translational equations by selecting the vehicle center of mass as the reference point. The vector form of the spacecraft attitude equations in terms of the body coordinates (x,y,z) is expressed as[3]

$$\frac{d}{dt} I\underline{\omega} + \underline{\omega} \times (I\underline{\omega}) = \underline{N}_d + \underline{N}_c \qquad (1)$$

where $I = diag(I_x, I_y, I_z)$ is the moments of inertia matrix and $\underline{\omega} = [\omega_x\ \omega_y\ \omega_z]^T$ is the body angular rate vector. $\underline{N}_d = [N_{dx}\ N_{dy}\ N_{dz}]^T$ is the disturbance torque vector and $\underline{N}_c = [N_{cx}\ N_{cy}\ N_{cz}]^T$ is the 3x1 attitude control torque vector from the reaction wheels, the magnetic coils, or the expansion jets.

The dynamic equation of the wheels can be expressed as

$$\frac{d}{dt}\underline{h} + \underline{\omega} \times h = -\underline{N}_c + \underline{N}_m \qquad (2)$$

where $\underline{h} = [h_x\ h_y\ h_z]^T$ is the vector of wheel momentum in body-fixed coordinates and $\underline{N}_m = [N_{mx}\ N_{my}\ N_{mz}]^T$ is the magnetic torque vector used for momentum dumping.

The 2-3-1 Euler angle sequence is chosen to transform the instantaneous body angular rates to Euler angular rates with θ_1, θ_2, and θ_3 to represent roll, pitch, and yaw angle, respectively. The spacecraft is assumed to be three-axis controlled that has one momentum wheel on each axis. Shown in figure 2 are the model spacecraft and the system coordinates. Also, the attitude rates are usually very small and in order to conveniently design the controller for attitude motion, Eq.(4) was linearized by neglecting the coupling terms and was simplified to three single-axis second-order differential equations

$$I_i\ddot{\theta}_i = N_{ci}, \qquad i = 1, 2, 3. \qquad (3)$$

Three control laws were designed and were implemented to compare the results between the platform and single PC. An optimum filter based on Kalman formulation was also designed to estimate the system states and to reduce the noise in the A/D signals.

a. PD Compensation

$$N_{ci} = K_{pi}\theta_i + K_{di}\dot{\theta}_i, \qquad i = 1, 2, 3. \qquad (4)$$

where θ_i is the Euler angle of 2-3-1 sequence, and K_{pi} and K_{di} are the proportional and the derivative gain, respectively. The control signal from the PD compensation, as shown by the formulation, is a direct multiple of the system states and will easily have some fluctuation whenever any noise exists in the feedback states.

b. Lead Compensation

$$N_{ci} = K_i \frac{T_f s + 1}{\alpha T_f s + 1}\theta_i, \qquad i = 1, 2, 3. \qquad (5)$$

where K_i is a constant gain, T_f is the equivilent time constant, and α is the zero/pole location ratio. To reduce the effects of noise to the controller, lead compensation is included which not only has fast response as the PD compensation does but also has the ability of filtering out high frequency noise.

c. LQR Compensation

$$J = \int_0^\infty \underline{x}^T Q \underline{x} + \underline{u}^T R \underline{u}, \quad i = 1, 2, 3. \tag{6}$$

$$N_{ci} = K_{opi}\theta_i + K_{odi}\dot{\theta}_i, \quad i = 1, 2, 3. \tag{7}$$

where $\underline{x} = [\theta_1\theta_2\theta_3\dot{\theta}_1\dot{\theta}_2\dot{\theta}_3]^T$ and K_{opi} and K_{odi} are the LQR gains derived from solving the associated Riccati equation. Although the control command by the LQR compensation is optimal with respect to the quadratic cost function, yet it has the same form as that by the PD compensation and shall have the same noise effect as well.

d. Filter Design

The filter includes two parts, the first part is for angle predition with angular rate measurements and the second part is for angle update with angle measurements[4]. The update frequency can be adjusted to suit the type of optical instrument used onboard. The filter can be expressed as

Predition:

$$\theta^-_{ik+1} = \theta^-_{ik} + (\dot{\theta}_{ik} + \dot{\theta}_{ik-1})\frac{T}{2} \tag{8}$$

Update:

$$\theta^+_{ik+1} = \theta^-_{ik+1} + K_{op}(\theta_{im} - \theta^-_{ik+1}) \tag{9}$$
$$i = 1, 2, 3, \quad k = 1, 2, 3, \cdots$$

where $\dot{\theta}_{ik}$ is the angular rate measurement, T is the period of the controller, θ_{im} is the angle measurement, K_{op} is the constant optimal filter gain, and k is the sequence number.

4. Modeling of Environmental Disturbances in Orbit

For low Earth orbit satellites, major attitude disturbances from the environment include gravity-gradient torque, magnetic disturbance torque, and aerodynamic torque. These three torques can be approximated by the following models[3]:

a. Gravity Gradient Torque

$$\vec{T}_G = 3\omega_o^2 \vec{R}_{o/c} \times (I \cdot \vec{R}_{o/c}) \tag{10}$$

where the gravity gradient torque $\vec{T}_G$ is caused by the unsymmetry of the spacecraft about the line $\vec{R}_{o/c}$, the position vector from the center of mass of the spacecraft to the center of Earth. ω_o is the orbital rate of the spacecraft about Earth, and I is the moment of inertia tensor of the spacecraft.

b. Magnetic Disturbance Torque

$$\vec{T}_M = \vec{M}_R \times \vec{B} \tag{11}$$

where the disturbance torque $\vec{T}_M$ is induced by the interactions between the residual magnetic field on the spacecraft, $\vec{M}_R$, and the magnetic field of Earth, $\vec{B}$. To simplify the model, the Earth's magnetic field is considered as a simple dipole with the magnetic "south" pole at 78.60° N latitude and 289.55° E longitude.

c. Aerodynamic Torque

$$\vec{T}_A = \vec{l}_{a/c} \times \vec{F}_{A/b} \tag{12}$$

where $\vec{T}_A$ is the aerodynamic torque, and $\vec{l}_{a/c}$ is the position vector from the center of mass to the aerodynamic center of the spacecraft, and $\vec{F}_{A/b}$ is the aerodynamic force vector, in terms of body-fixed coordinates, acting on the spacecraft.

For low Earth orbiting satellite, the disturbance torques are estimated in the order of 10^{-5} N-m.

5. Case Study

Consider a rigid model spacecraft in the Earth circular polar orbit with orbital height of 600 km. The moments of inertia of the spacecraft along body principal-axis are $I_x = 160kg - m^2$, $I_y = 119kg - m^2$, and $I_z = 140kg - m^2$. A three-axis simultaneous reorientation was commanded for the spacecraft to maneuver from the initial roll, pitch, and yaw angles of $\theta_x = 20^o$, $\theta_y = 20^o$, $\theta_z = -20^o$ to the final position of $\theta_x = 0^o$, $\theta_y = 0^o$, $\theta_z = 0^o$. The reorientation time shall be within 100 sec and shall be no overshoot in any of the angle displacement.

For the PD controller, the gains K_p = [−4.31 −4.52 −4.24] and K_d = [−81.3 −80 −81.45] were selected. The control parameters in lead compensation are K=-1.4, K2=-0.9, K3=-1.9 α = 0.01, T=21.4, 23.8, 21.5. The weighing matrices specified for LQR are Q=dia(5,5) and R=10 for

each axis and the associated optimal gains are K_{op} $[-0.707 \quad -0.707 \quad -0.707]$ and K_{od} $[-19 \quad -17.5 \quad -18.5]$. Because the they had been designed with the linearized system, the gains in each control method were tunned to obain expected responses when running the simulation with the nonlinear spacecraft model.

In the simulation, only the reaction wheels were used for attitude control. The frequency of the controller was adjusted at 22 Hz and the frequency of the dynamic system was about ten times of the controller frequency. To study the outputs of the controller, small estimation errors were created in the filter which were small for high frequency controller but became large for low frequency controller. The errors, with the characteristics of random noise, were then added to the angular rates and the estimated angles through estimation process.

Shown in figure 3 are the simulation results of the spacecraft attitude angles and the control commands for PD controller. The attitude motions from the platform are about the same as that from the single PC. However, the comparison with single PC shows the control commands sent out by the controller of the platform are rather disturbed. The noise in the filter are seeing to be directly affecting the control commands of the PD controller under slow control frequency. Angle and torque command trajectories for the Lead controller are shown in figure 4. Note that the simulation results from the platform matched very well with that from single PC. Also, with about the same angle motions, control commands from Lead controller are smaller than the commands from PD controller. Futher study indicates that reducing the frequency of Lead controller in the platform to 3 Hz and keeping the dynamics frequency unchanged will results in the same kind of disturbed control commands as from the PD controller.

The simulation results for the LQR controller are shown in figure 5. Similar attitude and control profiles like those from the PD controller are observed. The trajectories of the attitude angles are smooth for both single PC and the platform, but the trajectories of the torque commands from the LQR controller are rather disturbed. As can be expected the LQR is basically a PD controller with optimal gain coefficients. A comparison at the beginning of the torque commands, LQR controller shows the smallest initial torque commands among the three controllers for about the same angle trajectories.

6. Concluding Remarks

This study has established a PC-based simulation platform. To emulate the interactions between the controller and the dynamic system, the controller module and the dynamic module of the platform are placed in two different PC, with each running in parallel under independent speeds and exchanging data through A/D and D/A cards. A proper adjustment on the loop frequency of the dynamics software allows the platform to perform in real-time. As noted from the simulation results that the Lead compensation will be a better control method than the PD type of compensation methods for noise rejection.

In contrast to the single computer simulation, real-time simulation results indicate that the presented setup can be used for the preliminary design of a controller in an environment where the real-time software/hardware interaction effects are taken into account. Established with inexpensive equipment, the reported platform can conveniently and economically provide a more realistic environment for the researchers.

7. Acknowledgment

This work was supported by the National Science Council of R.O.C. under the project NSC85-2212-E-006-066.

8. References

1. Hampton, S.(1991), "PC -Based Simulation: An Altenative for Teaching Instrument Flying Skills," *ASME TS-Vol. 4, PC -Based Instrument Flight Simulation.*

2. Forster, S.(1991), "Technical Requirements for a Third-Generation PC -Based Instrument Flight Simulator," *ASME TS-Vol. 4, PC -Based Instrument Flight Simulation.*

3. Wertz, J.R. ed.(1978), *Spacecraft Attitude Determination and Control*, Kluwer Academic Publishers, Dordrecht, The Netherlands.

4. Grewal, M.S. and Andrews, A.P. (1993), *Kalman Filtering Theory and Practice*, Prentice Hall, Englewood Cliffs, New Jersey.

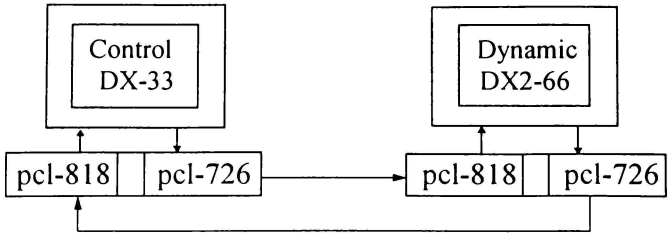


Figure 1 Schematic Diagram of Real-Time Simulation Platform

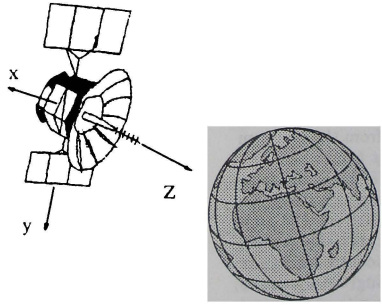


Figure 2 Model Spacecraft and the Body-Fixed Coordinates

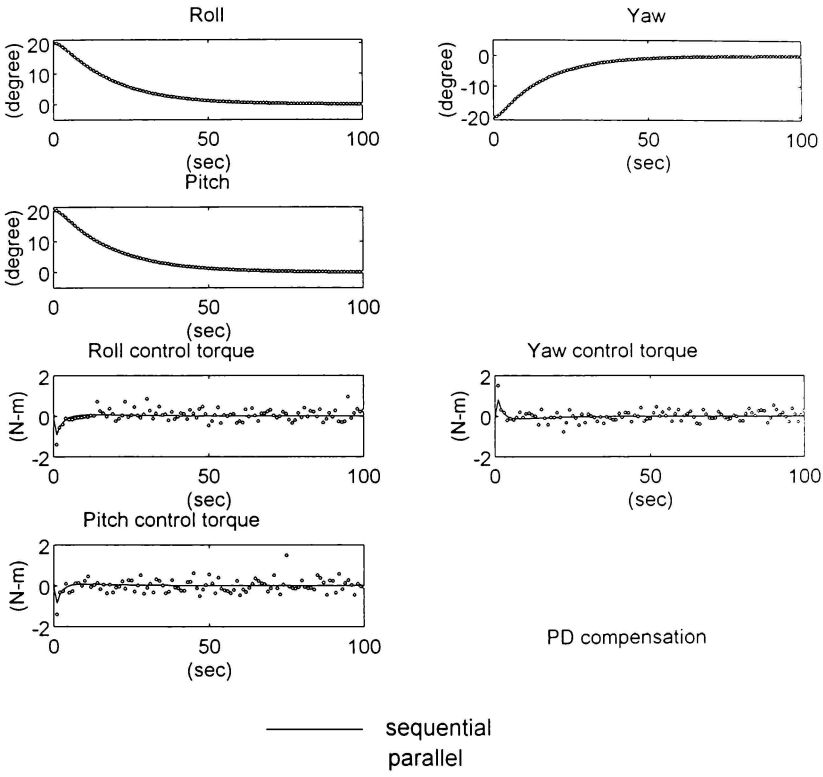


Figure 3 Spacecraft Attitude and Control Torques with PD Controller

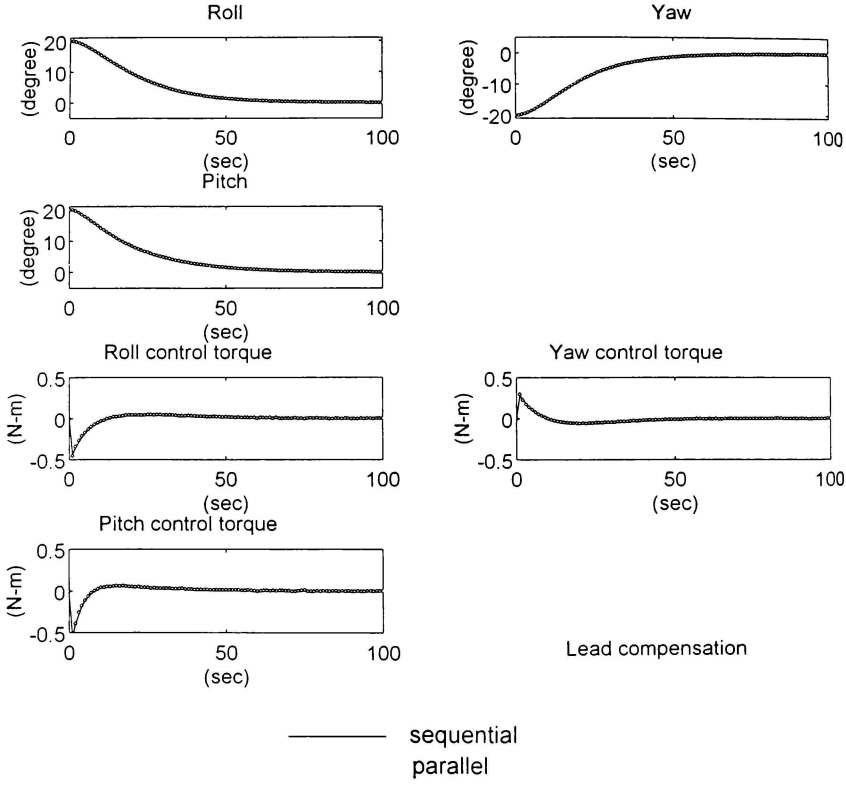


Figure 4 Spacecraft Attitude and Control Torques with Lead Controller

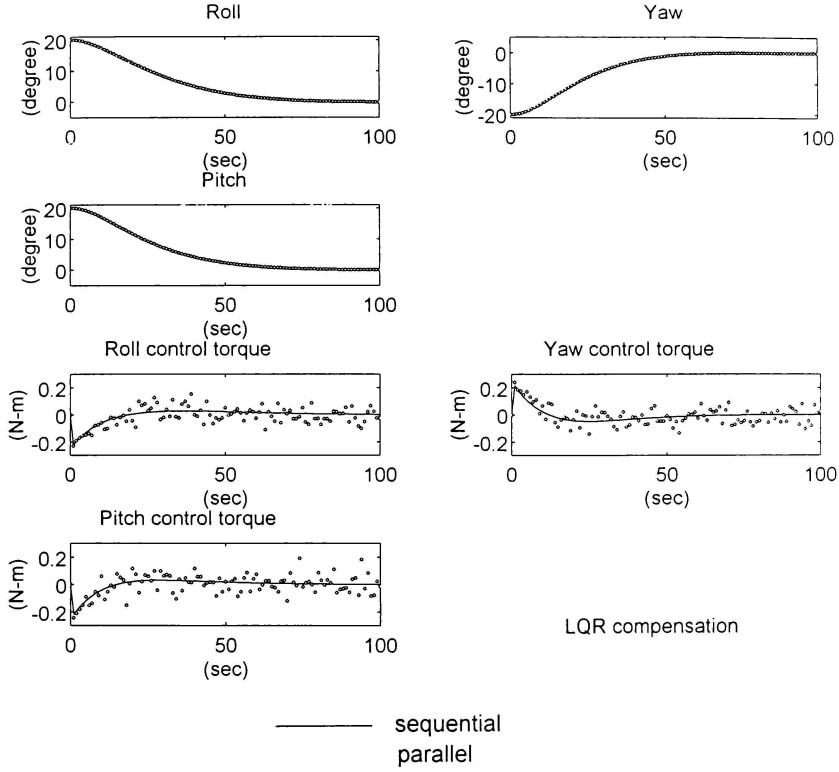


Figure 5 Spacecraft Attitude and Control Torques with LQR Controller

AIAA-97-3729

IDENTIFICATION OF DO-328 AERODYNAMIC DATABASE FOR A LEVEL D FLIGHT SIMULATOR

R. V. Jategaonkar* and W. Mönnich†
Institute of Flight Mechanics
German Aerospace Research Establishment (DLR)
Lilienthalplatz 7, 38108 Braunschweig, Germany

Abstract

Applying system identification methodology, a database for a high performance propeller aircraft of the type Dornier DO-328 with reversible flight controls was identified from flight data. The identified database including both rigid body aerodynamics and hinge moments caters to the stringent accuracy requirements of the Level D flight simulator for this transport aircraft. This paper provides an overview of the various aspects of database generation, emphasizing on the rigid-body aerodynamics. Some aspects of hinge moment modeling are also presented. The challenges encountered in development of the database are briefly brought out. The propeller effects and those at extreme flight conditions such as large steady sideslip and high angle of attack regimes including separated flow conditions of stall hysteresis on the rigid-body aerodynamics are demonstrated. As typical validation examples, the high activity tasks such as normal and crosswind landing as well as critical engine failure on takeoff are addressed. In all of these cases the end-to-end fidelity of the flight identified database has been ascertained through a comparison of the flight measured aircraft response with that predicted by an integrated model driven through force inputs.

Introduction

In general, nowadays the commercial transport aircraft of any size need to be supported by high-fidelity flight training simulators. The full-fledged flight-crew training simulators are often required to meet the Level D fidelity standards specified by the Federal Aviation Administration (FAA).[1,2] The fidelity of the flight simulation depends to a large extent on the accuracy of the aerodynamic database representing the aircraft, and it is generally the aircraft manufacturer's responsibility to provide an adequate database.

The flight training simulator for the DO-328 regional transport propeller aircraft with reversible flight controls manufactured by Dornier (now Fairchild-Dornier) incorporating the wind-tunnel predicted database showed certain limitations in the simulation fidelity. The aircraft manufacturer Fairchild-Dornier, hence, subcontracted the DLR Institute of Flight Mechanics to generate a suitable database meeting the FAA Level D requirements.

Although updating the wind-tunnel predictions through incremental coefficients obtained from flight data analysis is a viable approach,[3,4] the task of determining which terms in the aerodynamic model should be revised is formidable, the process highly iterative requiring considerable engineering judgement and can be as involved as identifying a new database from flight data. Moreover, the recent advances in aircraft modeling have led to analytical models for complex processes such as ground effects[5] and separated flow including stall hysteresis[6-8] making identification from flight data more amenable. Furthermore, parameter estimation algorithms enable processing of a large amount of flight data leading to a homogeneous database covering the entire operational envelope.[9-13] Based on these considerations and on the expertise available at DLR, it was considered appropriate to generate a new database from flight data applying modern methods of system identification rather than incrementally updating the wind-tunnel predictions. In the present investigations the output error method in the time domain is used to estimate the nonlinear models pertaining to the rigid-body and to the hinge moments. A stand-alone rigid-body aerodynamic database has been first identified from a model driven by measured aerodynamic control surface deflections, namely elevator, aileron, rudder, and spoiler deflections. The rigid-body aerodynamics includes the high angle of attack regime with stall hysteresis, large steady sideslip effects, ground effects, landing gear effects, and effects due to control surface malfunctions. The nonlinear influences due to the propeller slipstream were found to be predominant. In a

* Scientist, Senior Member AIAA.
† Scientist, Affiliate Member AIAA.
Copyright © 1997 by the Institute of Flight Mechanics, DLR. Published by the American Institute of Aeronautics and Astronautics, Inc., with permission.

separate step a hinge moment database for the above said primary controls has been identified from force driven dynamic models of the flight controls including static and Coulomb friction.

Subsequently, on the basis of end-to-end match[1,14] and without incorporating any closed-loop feedback drivers,[2] through an off-line simulation it was attempted to demonstrate the adequacy of the identified database not only for the 170 flight maneuvers defined in the Acceptance Test Schedule (ATS) covering only a part of the flight envelope, but for additional 1100 maneuvers at other flight conditions, thereby establishing the global validity. Because of the reversible flight controls, it was mandatory to demonstrate the end-to-end match based on the integrated model with force inputs (i.e. 6 degree-of-freedom equations of aircraft motion incorporating the identified rigid-body aerodynamic database coupled with the dynamic models of the flight control systems driven through pilot applied forces and incorporating the identified hinge moment database). This turns out to be significantly more complex than a model driven through control surface position inputs, because two highly nonlinear and complex systems, namely the rigid-body aerodynamics and the flight controls, are required to be modelled to a high degree of precision. Hence, demonstration of the end-to-end match based on the integrated model is rarely attempted by the suppliers of the aerodynamic databases. In this sense the investigations reported in this paper represent an unique attempt of generating the aerodynamic database suitable for a Level D flight simulator of DO-328 propeller aircraft with reversible flight controls.

At the time of writing this paper preparations for the FAA certification of the DO-328 flight training simulator including the identified database were under progress at Fairchild-Dornier, Oberpfaffenhofen, in Germany. This paper provides a brief description of the test aircraft, of flight tests carried out by the aircraft manufacturer Fairchild-Dornier, followed by an overview of the various aspects of database generation. Simulation fidelity of the identified database for a selected few critical cases has been demonstrated through a comparison of the flight measured aircraft response with the model predicted response.

DO-328 Aircraft and Flight Simulator

The Dornier 328, Fig. 1, is a high-wing, high-tail, twin turboprop regional transport aircraft designed to carry up to 33 passengers.[15] It is powered by two Pratt & Whitney Canada PW 119A turboprop engines with six bladed constant speed propellers, each capable of generating up to 2180 SHP. The aircraft has an overall length of 69 ft 10 in, wingspan of 68 ft 10 in, mean aerodynamic chord of 6 ft 8 in, wing area of 430.4 ft^2 and horizontal tail area of 97.2 ft^2. The maximum takeoff and landing weight is 27558 lb, rate of climb is 2420 ft/min and the maximum cruising speed is 335 KTAS. It has a maximum range of 700 nm with 30 passengers on board which can be more than doubled for half the number of passengers on board. In addition to the high speed capability at all altitudes, the DO-328 has a low final approach and landing speed providing short field takeoff and landing capability.[15]

All the primary flight controls are reversible, trimming is provided by trim tabs, and the rudder actuation is supported by a spring tab at airspeeds below 160 KIAS. The hydraulically operated roll spoilers are linked to the ailerons.

The DO-328 aircraft manufacturing program is supported by training facilities provided by six-axis motion based flight simulators at Oberpfaffenhofen in Germany, Fig. 2, and at Portland in USA. These flight simulators are to meet the FAA Level D fidelity allowing pilot training and check-outs to be carried out entirely on the simulator.[15]



Fig. 1: DO-328 Propeller regional transport aircraft

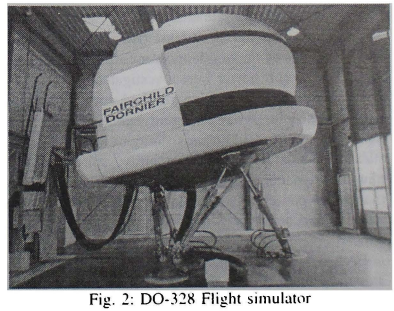


Fig. 2: DO-328 Flight simulator

Instrumentation System

The flight test instrumentation system provided measurement of the aircraft motion variables such as linear accelerations, angular rates, attitude angles, flow variables, and altitude above ground, of the control surface deflections for elevator, aileron, roll spoiler, rudder, ground spoiler and all the trim tabs including the spring tab, of the pilot applied control column forces, control wheel forces, and rudder pedal forces and the corresponding positions. All the above said measurements were obtained through dedicated sensors and signal processing units. In addition, the basic aircraft system provided the measurements of calibrated aircraft speed, total and static pressure, static temperature as well as of the engine states such as condition lever and power lever positions, propeller rpm, high pressure rpm, and torque for the left and the right engines. Furthermore, fuel quantity, brake pressures, wheel speeds, main gear and nose gear strut deflections, and stall warning were recorded. Typically, 142 signals including all of the above mentioned signals and a few others for check-out purposes were used for the data analysis. The sampling rate was 50 samples per second.

The angular accelerations and the control surface rates used to speed up the convergence of the parameter estimation algorithm were obtained through numerical differentiation of measured angular rates and surface deflections respectively in a data pre-processing step.

Flight Test Program

The flight test program for database generation was carried out by the aircraft manufacturer Fairchild-Dornier at Oberpfaffenhofen. Flight recordings for a total of 71 flights were provided for the database generation. Four different flap positions, namely 0°, 12°, 20°, and 32°, and for each flap setting at least two speeds and three power settings of flight idle, power for level flight, and maximum takeoff power were investigated. The in-flight maneuvers consisted of longitudinal trims, mistrims, climbs, descents, power change dynamics, gear change dynamics, control dynamics, longitudinal maneuver stability, static longitudinal stability, spiral stability, phugoid dynamics, and stalls. In addition, dynamic maneuvers particularly suitable for system identification were carried out.

Each system identification maneuver was initiated from a trimmed level flight. The maneuver sequence consisted of an elevator doublet input exciting the short period motion, a large amplitude elevator pull and push, a bank-to-bank maneuver and an aileron step input exciting the rolling motion, a rudder doublet input exciting the dutch roll, and a rudder step input resulting in steady sideslip conditions. Steady sideslip maneuvers with angle of sideslip up to 12° in either direction were carried out with retracted and extended landing gear. Several fly-bys over the runway at different altitudes of down to 10 ft above ground were recorded to enable identification of ground effect.

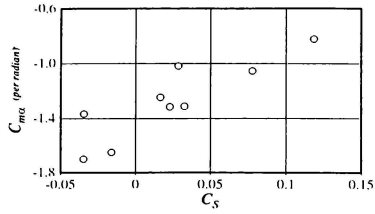
Although the bulk of the data analyzed was for nominal position of 25% center of gravity, tests were also carried out with forward and aft center of gravity locations. Furthermore, tests were carried out with both left and right lateral center of gravity which affects the aileron angles required for trim.

The tests included acceleration and deceleration on ground with and without reverse thrust, normal takeoff and engine failure on takeoff, normal landing and crosswind landing. The normal in-flight, dynamic input maneuvers and the flights in ground effect were repeated for the four flap settings with one engine inoperative. All system identification and validation test maneuvers were flown manually by the test pilot.

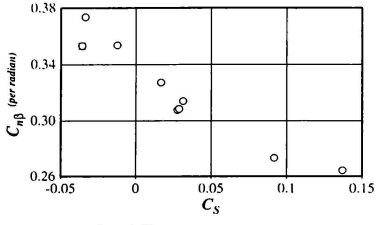
Rigid-Body Aerodynamic Database

The rigid-body aerodynamics and hinge moments are identified separately mainly because the two databases are well separable. This separation leads to somewhat smaller size and more tractable models for parameter estimation. The turn around time in such a case is comparatively smaller, which is very important during the iterative phase of model building. Moreover, it also eliminates an adverse influence of any inaccuracies in the hinge-moment database on the estimation of rigid-body aerodynamics and vice-versa.

Identification of an aerodynamic database has been carried out by postulating suitable derivative models in an analytical form. Estimation of a comprehensive aerodynamic model is an iterative process, which starts with point-identification at all of the points flight tested. Point-identification from small excursion maneuvers around a trim point enables estimation of primary derivatives without accounting for Mach number, thrust setting or landing flap effects and results in a model related to a specific trim condition. For example, Figs. 3a and 3b show respectively the estimates of the longitudinal static stability, derivative $C_{m\alpha}$, and of the



a) Longitudinal static stability



b) Weathercock stability

Fig. 3: Estimates from point identification

weathercock stability, derivative $C_{n\beta}$, for the clean configuration obtained from the analysis of dynamic maneuvers. It is obvious that the thrust coefficient C_S has a very strong influence on these primary derivatives. In general it was found that the thrust setting affects significantly the aircraft motion in all the axes.

Collating the point-identification results the structure of the aerodynamic model valid over a broader range was determined. The model incorporating linear as well as nonlinear dependencies due to angle-of-attack, Mach number, angle of sideslip, and thrust-coefficient was postulated. Based on such a model the aerodynamic derivatives are estimated separately for the four flap positions. In each case several flight maneuvers were concatenated in a multi-point identification to yield a single set of parameters for the symmetrical flight configuration with two engines on and landing gear retracted. Typically, for each flap position 70 to 80 flight maneuvers were analyzed to estimate about 60 derivatives pertaining to the longitudinal mode and more than 100 parameters pertaining to the lateral-directional motion from another 80 to 90 maneuvers. In the following step, the landing gear and ground effects were identified as incremental terms. The model is then extended to include stall hysteresis. This basic aerodynamic model is then augmented with effects due to single engine operation. It was found that the single engine effects are strongly asymmetric. In terms of aerodynamic derivatives, roughly 350 derivatives were identified per flap position to arrive at a comprehensive rigid-body aerodynamic model. Detailed description of the complete modeling is not within the scope of this paper, however, a typical example of stall hysteresis is briefly elaborated here.

From quasisteady stall maneuvers an unsteady aerodynamic model for high lift including flow separation and stall was identified. Based on the Kirchhoff's theory of flow separation, the wing lift can be modeled as a function of angle of attack α and flow separation point X:[6-8]

$$C_L(\alpha, X) = C_{L\alpha}\left(\frac{1+\sqrt{X}}{2}\right)^2 \alpha \qquad (1)$$

where $C_{L\alpha}$ is the lift curve slope and X ($0 \le X \le 1$) is a dynamic variable describing the instantaneous location of the flow separation point along the chord on the upper surface of the wing. X = 1 and X = 0 correspond to attached and fully separated flow respectively. Considering simplified expressions for the purpose of demonstration, the total drag and pitching moment coefficients are modeled as:[6]

$$C_D = C_{D0} + \frac{1}{e\,\pi\,\Lambda} C_L^2(\alpha, X) + \frac{\partial C_D}{\partial X}(1 - X) \qquad (2)$$

$$C_m = C_{m0} + C_{m\alpha}\,\alpha + C_{m\delta e}\,\delta_e + \frac{\partial C_m}{\partial X}(1 - X) \qquad (3)$$

where δ_e is the elevator deflection, e the Oswald factor and Λ the aspect ratio of the wing. The drag polar using $C_L(\alpha, X)$, the lift modified due to flow separation, automatically accounts for the major contribution to the unsteady drag. Additionally, empirical correction terms $\partial C_D/\partial X$ and $\partial C_m/\partial X$ proportional to the flow separation were incorporated to account for any additional effects.

The flow separation point X may be determined from wind-tunnel tests or can also be identified from flight test data using the approximation:

$$X = \frac{1}{2}\left\{1 - \tanh\left|a_1\left(\alpha - \tau_2\,\dot{\alpha} - \alpha^*\right)\right|\right\} \qquad (4)$$

where $\dot{\alpha}$ is the rate of change of angle of attack, τ_2 the time constant accounting for the unsteady aerodynamic effects, α^* the breakpoint corresponding to X = 0.5, and a_1 the static stall characteristics of the airfoil. The time constant τ_2 depends on airfoil section and wing

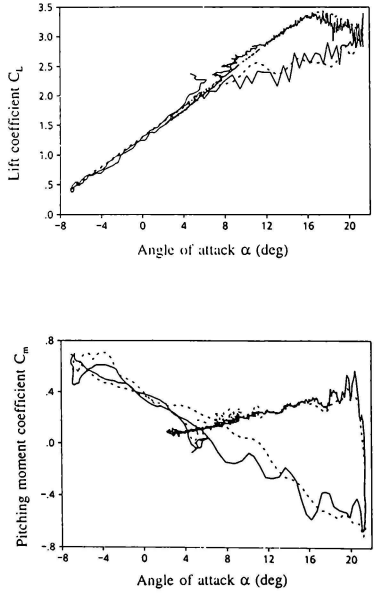
configuration. The three parameters a_1, τ_2, and α^* are completely adequate to model the stall hysteresis. Although not considered in the present investigation, it is also possible to model time-dependent flow separation based on an approximation of the Wagner or the Theodorsen function.[6,9]

Fig. 4 shows the modeling and simulation of aircraft stall based on the rigid-body stand-alone model (i.e. 6 degree-of-freedom model driven through flight measured control surface deflections and incorporating the identified database). Figs. 4a shows a comparison of the lift coefficient C_L and of the pitching moment coefficient C_m obtained from measured flight data (shown by solid lines) with those identified in the database (shown by dashed lines). Fig. 4b shows a comparison between the flight measured and model predicted time responses for α the angle of attack, a_z the vertical acceleration, q the pitch rate, θ the pitch attitude, and V the true airspeed. Adequacy of the identified model in terms of the FAA-specified tolerances is apparent from the time history plots and the efficacy of the analytical model postulate is evident from the hysteresis in the lift and the pitching moment coefficients.

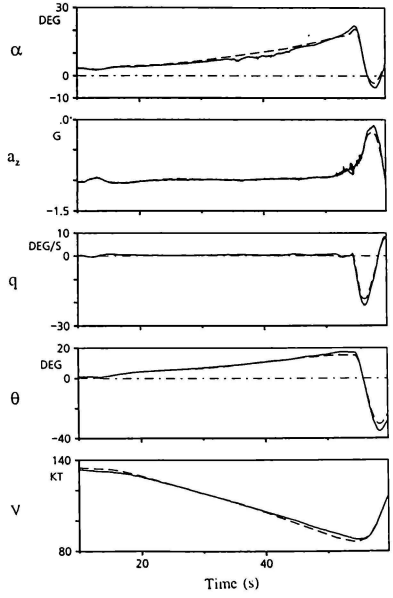
Accurate stall modeling would lead to an effective flightcrew training on a simulator under extreme flight conditions with flow separation, and meets the recent recommendations made by the NTSB (National Transportation Safety Board) after a Jetstream 4101 aircraft crash following a stall due to improper pilot reaction to stall warning.[16]

Hinge Moments Database

The same set of flight maneuvers used to estimate the rigid-body aerodynamics was analyzed to derive the hinge moments database. The same philosophy of point-identification leading to model structure determination followed by multi-point identification from a large set of concatenated maneuvers was adopted in this case too.



a) Aerodynamic Coefficients



b) Comparison flight measured and estimated responses

Fig. 4: Modeling and Simulation of aircraft stall (——— Flight measured, - - - - - Model predicted)

It is noteworthy that for all the flight controls a one degree-of-freedom system with two state variables is sufficient as basic dynamic model. As state variables the left elevator deflection (the left and the right elevator being mechanically coupled), the right aileron deflection (the left aileron being kinematically coupled to the right), the rudder deflection, and the corresponding rates are chosen for the three flight control models. All the inertias (pilot effector, linkages and surface flaps) are lumped at the respective control surface axis of rotation. Friction and aerodynamic damping contribute to the total system damping. The spring forces are provided by the aerodynamic hinge moments, in the case of elevator additionally by a mechanical downspring and in the case of stall by a stick pusher. The inputs to the dynamic model are the pilot applied forces that are reduced to moments about the control surface axes via kinematic relationships and the inertia moments resulting from aircraft motion. Outputs are control deflections, deflection rates, and pilot effector deflections. The roll spoilers are linked to the ailerons with no additional dynamics, but having a deflection rate limit on extension.

The primary contributions to the hinge moments stem from control surface deflections, the respective tab deflections, and the local flow angles. Particularly for the rudder, asymmetrical effects due to the thrust coefficients were very important. Mach number and landing flap dependencies were of secondary importance. In addition to the hinge moments, some mechanical properties such as static and Coulomb friction coefficients, deflection limits, and cable stretch coefficients had to be identified.

Fig 5 shows typical results obtained from the dynamic models of the flight controls for two flight maneuvers with steady sideslip, where δ_{eL} is the left elevator deflection, δ_{aR} is the right aileron deflection, δ_r is the rudder deflection, f_{EC} is the control column force, f_{AC} is the wheel force, and f_{PC} is the pedal force. The fidelity of the stand-alone hinge moment database is apparent from the match between the flight measured control surface deflections shown by solid lines and the model predicted deflections shown by dashed lines.

A major problem caused by friction was encountered during identification of the hinge moment database applying the gradient based Modified Newton-Raphson optimization method.[9] Due to the strong nonlinearities in the static friction the numerically determined gradients can be inaccurate, leading to abrupt termination of the optimization due to singular information matrix, to convergence problems, and to local minima or solutions that are not even local minima. A satisfactory solution to this problem could not be found within the course of the DO-328 database development, hence, in several of these cases 'hand-optimization' was necessary.

Models for Identification and Simulation

The identification of the rigid-body aerodynamics and of the hinge moments carried out based on the analytical model postulates yields four sets of derivatives for four flap positions which are finally combined in a single database incorporating a linear interpolation procedure for flap position only. This so-called 'Derivative-model' is, however, rarely used in the flight simulators mainly because the database in the so-called 'Table-model' form was hitherto mostly generated from wind-tunnel measurements. Following the conventional way and in the present case as desired by the aircraft manufacturer, the flight identified database has been converted into a Table-model. In this form the rigid-body aerodynamics database consists of 124 tables out of which 14 are five-

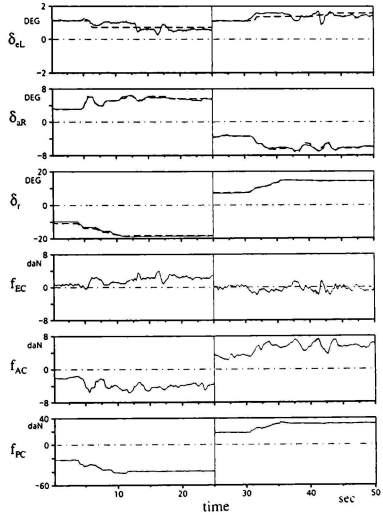


Fig. 5: Fidelity of stand-alone hinge moment database (—— Flight measured, - - - - - Model predicted)

dimensional, 28 are four-dimensional, 45 are three-dimensional, 27 are two-dimensional, and 10 are one-dimensional functions. The elevator, aileron, and rudder hinge moments required just 8, 18, and 27 tables respectively.

Although the Derivative-model is better suited for parameter estimation, keeping track of a large number of derivatives being estimated is not easy, particularly those representing the secondary effects. Furthermore, over parametrization of the model, although leading to better response match, is detrimental to identifiability of parameters and can lead to unrealistic estimates. The Table-model, or at least the plots of the identified aerodynamic coefficients, are more amenable to physical interpretation of the effects they are to represent.

The two alternative model forms, i.e. the Derivative-model and the Table-model, are equivalent in terms of the accuracy, but the Derivative-model is significantly smaller in size and computationally faster than the Table-model because the linear-interpolation is reduced to that for just one independent variable. The stand-alone rigid-body aerodynamics in the Table-model form required about 8 times the computational time compared to that for the Derivative-model, the ratio becoming as large as 12-14 for the integrated model using both the rigid-body aerodynamics and the hinge moments. Although as realized in the present case, implementation of computationally complex large size databases is feasible with modern computers, the differences in the two approaches are significant and pave way to reconsider the alternative way of implementing the flight data identified models in flight simulators.

Validation Examples

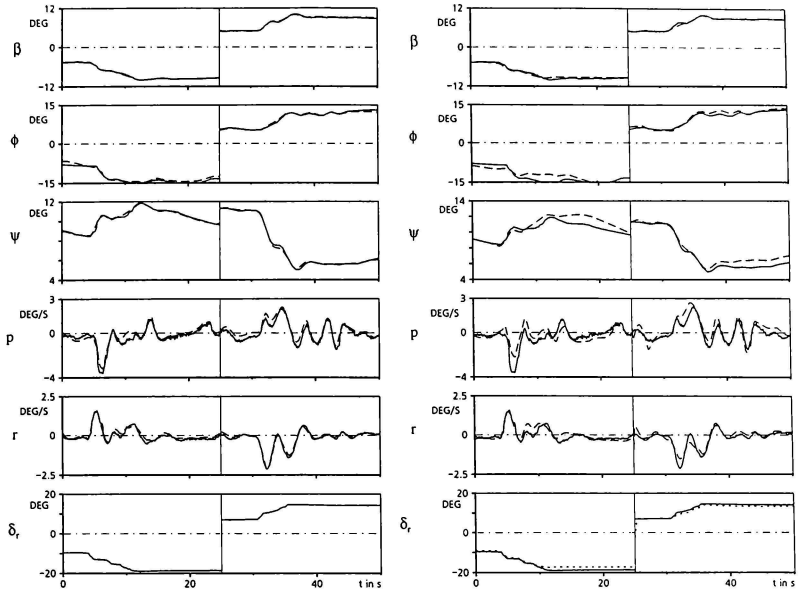
As already pointed out in the Introduction, apart from the 170 flight maneuvers specified in the ATS covering just a part of the flight envelope, a large number of in-flight maneuvers, maneuvers in ground effect at various altitudes above ground for different power settings under symmetric and asymmetric (one-engine) flight conditions were evaluated to establish global validity of the model. A few selected critical examples which bring out the different features of the database are presented here.

Steady State Sideslip Maneuver

The first example pertains to flight maneuvers at large sideslip angles and serves the purpose of exhibiting correct inter-relationship of steady state lateral/directional flight characteristics. Fig. 6 shows typical variables pertaining to the lateral-directional motion for two time segments with negative and positive angles of sideslip for approach configuration of 20° flaps with landing gear down; β is the angle of sideslip, p is the roll rate, r is the yaw rate, ϕ is the bank angle, ψ is the true heading, and δ_r is the rudder deflection. Fairly large rudder deflections of about 20° were applied resulting in steady sideslip up to 11°. Each maneuver is of 25 s duration and covers both the transition phase from one level of sideslip to another as well as trim phases. Fig. 6a shows the match between the flight measured and model predicted responses for the stand-alone rigid body model (i.e. six degree of freedom equations of aircraft motion excited through the flight measured control surfaces including the identified rigid-body aerodynamics). The exceptionally good match in this case and also that in Fig. 4 for the stall maneuver demonstrate the excellent fidelity of the rigid-body aerodynamic database.

For the same case, Fig. 6b shows the end-to-end match with the integrated model (i.e. six degree-of-freedom equations of aircraft motion coupled with dynamic models of the flight controls incorporating identified databases for rigid body and hinge moments). The distinction between the two evaluations shown in Figs. 6a and 6b is apparent from the bottom most plot, which shows just the measured rudder deflection in Fig. 6a as input to the model, whereas in Fig. 6b the model predicted rudder deflection is compared with the flight measurement. In Fig. 6b the computed control deflections are then the inputs to the rigid-body aerodynamics. Although some deterioration in the end-to-end match is observed in Fig. 6b compared to Fig. 6a, the overall match between flight measured and model predicted variables is good and within the FAA-tolerances of ± 2° for bank angle and ± 1° for sideslip; the aileron deflection not shown here was within ± 10% or ±2° and the wheel force bias within ± 1.3 daN and the pedal force bias within ± 2.2 daN. It may be pointed out that it was necessary to account for asymmetric propeller effects, particularly on the vertical tail, at such extreme flight conditions.

The deterioration of the match in Fig. 6b compared to Fig. 6a clearly brings out the difficulties faced by the database developers in demonstrating the end-to-end match obtained by combining two highly complex models, although each has an excellent fidelity as evident from Fig. 6a and Fig. 5. Any attempt to further refine the stand-alone models appear unrealistic. In the worst case and if critical, some tuning may be possible using the integrated model.



a) Stand-alone rigid-body aerodynamics

b) End-to-end match with integrated model

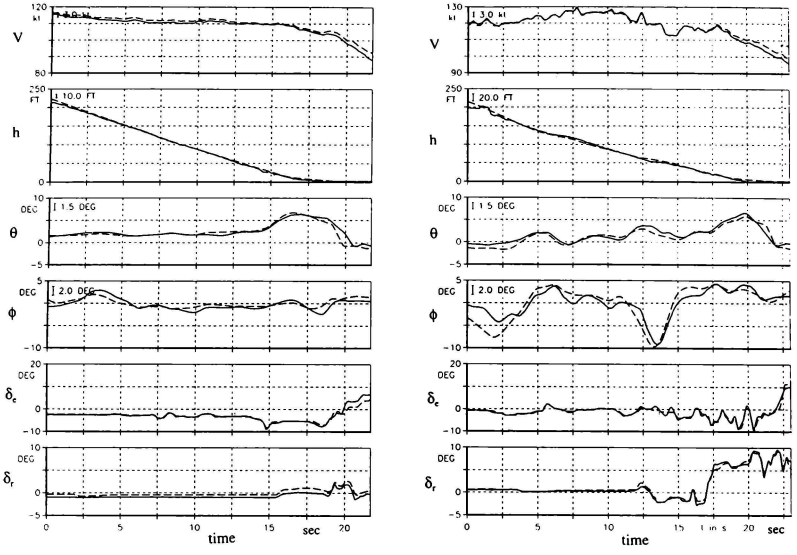
Fig. 6: Simulation fidelity for steady sideslip maneuvers at approach configuration with extended landing gear (——— Flight measured; - - - - - Model predicted)

Normal and Crosswind Landing

The second example pertains to normal and crosswind landing and demonstrates that the landing characteristics during this high activity task as reproduced in the database conform to the airplane.

Figs. 7a and 7b show the proof-of-match for the normal and the crosswind landing respectively. As a typical case the results for 20° flaps are presented here. Starting from 200 ft above ground, the approach, flare, touch down, derotation, and initial ground roll is simulated as a single maneuver, though FAA allows to split up the landing phase into two segments (approach and derotation after touchdown). Furthermore, it is also worth to point out that the simulation has been carried out without closed-loop controllers, although feedback-drivers on, for example, pitch angle (with elevator or control column) and bank angle (with control wheel) are usually applied.[2] In both the cases of normal and crosswind landing the calibrated airspeed V is within the specified tolerance of ± 3 kts, altitude h within ± 10 ft, pitch angle within ± 1.5°, and bank angle mostly within ±2° (the applicable tolerances are shown by vertical bars in the figure). The column force was within ± 2.2 daN of the measured value. Fig. 7 also shows the time histories of the elevator and rudder deflections. Except for some small deviations in the elevator match after touch down in Fig. 7a, the overall match for the control surface deflections obtained from the force driven models is also good.

The simulation of the phase on and close to the ground is influenced by the ground effect and the wind.[17] The identification of ground effects, although not elaborated in this paper, was carried out from the fly-by maneuvers. Analytical model postulates incorporating a ground effect function approximated by tanh-function



a) Normal landing

b) Crosswind landing

Fig. 7: Proof-of-match for high activity task of normal and crosswind landing
(——— Flight Measured; - - - - - Model predicted)

provided a convenient way to account for ground effect and was directly amenable for parameter estimation.[5] Furthermore, it results in a continuous transition from in-flight regime to on-ground and thereby leads to a single database covering both the regimes.

In the majority of the in-flight cases it was not necessary to account for the wind. For several of the flights in ground-effect, takeoffs, and landings it was adequate to account for flight-card noted wind velocity and wind direction. It does not, however, account for the turbulence. Hence, for those cases in which a high level of turbulence is encountered and for crosswind cases, the wind components (u_{wg}, v_{wg}, w_{wg}) are obtained as the difference between the measured tracking speed (i.e. north-south velocity, east-west velocity, and rate of climb) provided by the IRS (Inertial Reference System) and the components of the true airspeed transformed in the earth-fixed coordinate system.

Critical Engine Failure on Takeoff

The third example pertains to the simulation of the aircraft response to an engine failure during the critical phase of lift-off, the response to rudder and aileron being of particular interest. Fig. 8 shows the simulation results for a few pertinent variables. The complete sequence starting from stand-still position, acceleration, rotation, and climb to 200 ft above ground is simulated as a single maneuver. As in the previous case of landings, the simulation results were obtained without any closed-loop control. From the bottom most plot in Fig. 8 of F_L and F_R, the thrust due to the left and the right engine, it is observed that the critical left engine is shut down immediately after rotation. To maintain the aircraft bank and heading the pilot applies about 15° rudder (δ_r in Fig. 8) and about 6° ailerons not shown in the figure. Except for a small delay in the bank angle, which is still within the tolerance, the match for all the variables is good. Specifically, the true airspeed is within the

tolerance of ± 3 kts, altitude within ± 20 ft, pitch angle within ± 1.5° and the bank angle within ±2°, and thus the database meets the fidelity requirements specified by the FAA.

For the majority of the ATS and the additional tests cases the quality of the match was the same as that of examples presented in this paper. Only a few cases pertaining to ground roll and roll response due to large rudder inputs during single engine operation covered in the additional validation tests showed some deterioration in match for a few variables. However, considering the complexity of the rigid-body aerodynamics and that of the hinge moments for the propeller aircraft with reversible controls, in the overall sense it was still considered acceptable.

Concluding Remarks

The investigations presented in this paper provide an overview of a project which was successfully completed in the most recent past. It led to the development of a six degree-of-freedom model of aircraft motion with a very complex rigid-body aerodynamics of the DO-328 propeller aircraft and to dynamic models for reversible flight controls including friction and hinge moments. The approach adopted to arrive at a global model is elaborated. The nonlinear rigid-body aerodynamics model is strongly influenced by the propeller slipstream and includes the high angle of attack regime including stall hysteresis, large steady sideslip effects, ground effects, landing gear effects, and effects due to control surface malfunctions. The simulation fidelity of the flight-identified database has been demonstrated on a few interesting cases of stall, large steady sideslip, normal and crosswind landing, and engine failure on takeoff. The fidelity of the stand-alone rigid body aerodynamics database is found to be exceptionally good. The end-to-end match based on the integrated model driven through force inputs for an aircraft with reversible flight controls is within the tolerances specified for a Level D flight simulator. Global validity of the database has been demonstrated on a very large number of flight maneuvers covering the entire operational flight envelope.

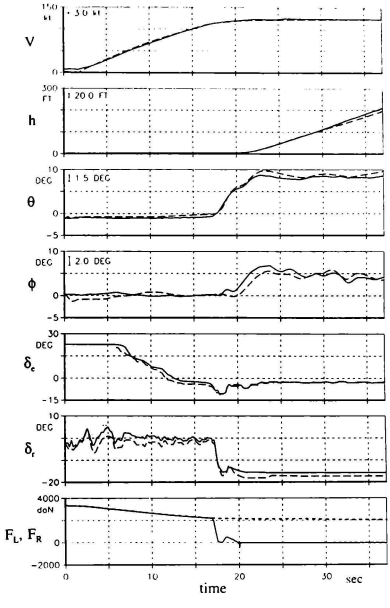


Fig. 8: Proof of match for critical engine failure on takeoff (——— Flight measured; ---- Model predicted)

Acknowledgements

Inter-divisional team effort had led to the development of the DO-328 database and to the generation of the proof-of-match data. Apart from the authors of the paper the following individuals from DLR contributed to the successful completion of this challenging task: D. Altenkirch, D. Fischenberg, Dr. W. Gockel, W. Heine, Dr. B. Krag, Dr. E. Plaetschke, D. Rohlf, M. Rohlfs, L. Thiel, and S. Weiss. The authors would like to express their appreciation to Mr. G. Fischer, Mr. A. Straile and Mr. T. Otte of Dornier GmbH, Germany for several hours of useful discussions on the various aspects of flight data supplied by Dornier, engine model, and measurement system. The authors are also thankful to the aircraft manufacturer Fairchild-Dornier for having granted the permission to publish the investigations reported in this paper.

References

[1]Anon., "Airplane Simulator Qualification," FAA Advisory Circular, AC 120-40B, Interim Version, Feb. 1991.

[2]Anon., "Airplane Flight Simulator Evaluation Handbook," The Royal Aeronautical Society, London, Feb. 1995.

[3]Neville, K. W. and Stephens, A. T., "Flight Update of Aerodynamic Math Model," AIAA Flight Simulation Technologies Conference, Paper No. 93-3596, Monterey, CA, Aug. 9-11, 1993.

[4]Trankle, T. L. and Bachner, S. D., "Identification of a Nonlinear Aerodynamic Model of the F-14 Aircraft," *Journal of Guidance, Control, and Dynamics*, Vol. 18, No. 6, Nov.-Dec. 1995, pp. 1292-1297.

[5]Fischenberg, D., "Identifizierung des Bodeneffekts und der Fahrwerksaerodynamik des ATTAS aus Flugversuchsdaten" (In German), DLR IB 111-94/22, June 1994.

[6]Fischenberg, D., "Identification of an Unsteady Aerodynamic Stall Model from Flight Test Data," AIAA Atmospheric Flight Mechanics Conference, Paper No. 95-3438, Baltimore, MD, Aug. 7-10, 1995.

[7]Goman, M. and Khrabrov, A, "State-Space Representation of Aerodynamic Characteristics of an Aircraft at High Angles of Attack," *Journal of Aircraft*, Vol. 31, No. 5, Sept.-Oct. 1994, pp. 1109-1115.

[8]Leishman, J. G. and Nguyen, K. Q., "State Space Representation of Unsteady Airfoil Behavior," *AIAA Journal*, Vol. 28, No. 5, May 1990, pp. 836-844.

[9]Hamel, P. G. and Jategaonkar, R. V., "Evolution of Flight Vehicle System Identification," *Journal of Aircraft*, Vol. 33, No. 1, Jan.-Feb. 1996, pp. 9-28.

[10]Baillie, S. W., Hui, K., and DeLeeuw, J., "The Flight Test and Data Analysis Program for the Development of a Boeing/deHavilland Dash 8 Simulator Model," CP-519, AGARD, Oct. 1992, Paper No. 30.

[11]Mulder, J. A., Baarspul, M., Breeman, J. H., Nieuwpoort, A. M., Verbraak, J. P. T., and Steeman, P. S. J. M., "Determination of the Mathematical Model for the New Dutch Government Civil Aviation Flying School Flight Simulator," *Proceedings of the 18th Annual Symposium of the Society of Flight Test Engineers*, Amsterdam, The Netherlands, 1987, Paper No. 15.

[12]Jategaonkar, R. V., Mönnich, W., Fischenberg, D., and Krag, B., "Identification of C-160 Simulator Data Base from Flight Data," *Proceedings of the 10th IFAC Symposium on System Identification*, Copenhagen, Denmark, Elsevier Sciences Ltd., IFAC Publications, Oxford, UK, 1994, pp. 1031-1038.

[13]Jategaonkar, R. V., Mönnich, W., Fischenberg, D., and Krag, B., "Identification of Speed Brake, Air-drop, and Landing Gear Effects from Flight Data," *Journal of Aircraft*, Vol. 34, No. 2, March-April 1997, pp. 174-180.

[14]Neville K. W., "Simulator-to-Flight Validation - Methods, Problems and Dilemmas," RAeS Symposium on 'Data Issues for Flight Simulators - An On-Going Problem', Paper No. 14, London, UK, Nov. 1993.

[15]Anon. Dornier-328 (Product Information), Dornier Luftfahrt GmbH, Germany, March 1992.

[16]Ray, P. A., "Federal Aviation Administration Flight Simulation Standards - A Regulatory Update," AIAA Flight Simulation Technologies Conference, Paper No. 95-3389, Baltimore, MD, Aug. 7-10, 1995.

[17]Fischenberg, D., Mönnich, W., Krag, B., and Jategaonkar R. V., "Aspects of C-160 Simulator Model Determination and Validation On and Close to the Ground," AIAA Flight Simulation Technologies Conference, Paper No. 94-3404, Scottsdale, AZ, Aug. 1-3, 1994.

CONCEPTS FOR DETECTING PILOT-INDUCED OSCILLATION USING MANNED SIMULATION

Ba T. Nguyen[‡], Thomas J. Cord[§]
Wright Laboratory, Wright-Patterson AFB, Ohio, 45433

Abstract

A 1985 USAF Test Pilot School project, HAVE PIO, performed a flight evaluation of the longitudinal pilot-induced oscillation (PIO) tendencies of several representative aircraft configurations. The evaluations have been duplicated on ground-based simulators using the same precision offset landing task. Comparisons are presented based on Cooper-Harper ratings, PIO ratings, time history data and pilot comments. Trends are established and an evaluation of the concepts proposed to make ground-based simulators a more effective tool for detecting PIO are presented.

Acknowledgments

The authors wish to acknowledge the contributions of the following individuals: Mr. Brian K. Stadler from WL/FIGD and Mr. Gary Jeff Slutz from Halifax Corporation at Wright Patterson AFB for their implementation of the Have PIO model on the simulators and their valuable suggestions regarding test setup and conduct, Mr. David G. Mitchell of Hoh Aeronautics Inc. (HAI) for his analysis of the simulation and flight test data and his many thoughts on how the two evaluation techniques relate, Dr. Dominick Andrisani of Purdue University for his creation of the first version of the Frequency Investigator and Detector Oscillation (FIDO) program which we used for the time history analysis, Mr. Joseph A. Baker of ASC/EN for his modification of FIDO to improve the correlation among the FIDO results, pilot comments and engineer's visual inspection of the time histories, Lt. Lori Ann Foringer of the flying qualities group for her assistance during the piloted simulation, Mr. David B. Leggett of the flying qualities group for his constant efforts to make the final results of high quality. We would also like to acknowledge the pilots who flew the tests over and over and over: Majors Duffy, Glade, Nolan, and Hunnell from the F-22 SPO, Lt/Col Sasscer and Major Hunter from the Training SPO, Majors Esch and Jasper from the F-16 SPO, and Major Smith from the JAST program.

Introduction

Ground-based simulators have become an indispensable tool in the development of modern aircraft. By providing piloted evaluations of aircraft flying qualities long before the aircraft is ready to fly, simulators allow designers to detect problems, redesign, and re-evaluate aircraft early in the design process. This is an enormous advantage, since flight control fixes become more expensive and have greater impacts on schedule as the development process continues. For PIOs however, experience has shown that ground-based simulation has not been particularly successful at detecting problems prior to flight test. It should be noted that in many programs, PIOs encountered in flight test have been subsequently duplicated in the simulator. This gives some confidence that prediction of PIO in early simulation may be possible.

The purpose of the study presented here is to develop a modified ground-based simulation method which will improve the likelihood of early PIO detection. The HAVE PIO flight test program in reference [1] was used as the truth model. In this flight test program, 18 configurations were flown in the landing phase on the NT-33A variable-stability aircraft. All of the PIO problems were due to linear causes (excessive phase lag and low responsiveness) rather than rate limiting or mode switching. The PIO tendencies of these configurations ranged from none to severe. The wide range of tendencies was important to preclude an evaluation process which would result in false PIO predictions

The effort consisted of two phases. The purpose of Phase I was to establish a baseline comparison between ground-based simulation and flight. The purpose of Phase II was to find the methods to improve the correlation between the two.

This paper is declared a work of the U.S. Government and is not subject to copyright protection in the United States

[‡]Flight Control Engineer.
[§]Aerospace Engineer.

Phase I data analysis was completed in 1996 and was presented in reference [2] at the AIAA AFM conference in San Diego. As expected, the simulation evaluation of PIO tendency did not match the findings of flight test. However, the simulations did show the same general trends as seen in flight. The best configurations in flight tended to be the best configurations on the simulator. Mediocre flight configurations also tended to be mediocre on the simulator although they were a little better on the simulator than in flight. (An alternate interpretation is that configurations predicted to have Level 2 flying qualities on the simulator ranged in flight from Level 1 to Level 3.) The worst configurations in flight were the worst configurations on the simulator, but were not as severe by a significant margin.

This paper addresses a) the effects of both simulation and task changes on the ground-based simulation pilot ratings and b) time-history data analysis. Due to pilot availability at a time of Phase II simulation, only one of the four Phase I evaluation pilots (D) was able to participate in Phase II. All pilots are test pilots with varied backgrounds, as shown in Table 1.

Table 1 Test Pilot Experience (hours)

	A	B	C	D
F-15	1000	400	800	-
F-16	-	-	500	1017
F-18	-	-	350	-
F-111	-	1100	-	-
A-7	-	200		-
T-38	1000	150	800	-
B-52	-	-	-	1756
B-2	-	-	-	126

Flight Test Descriptions

The HAVE PIO flight test program was well documented in References [1] and [2]. In addition, the flight test time histories have been recovered and are available from the Flight Dynamics Directorate of Wright Laboratory. The task performance data was not recorded, thus making a complete comparison of the two data bases impossible.

PIO Rating Scales

Two PIO rating scales were used in the HAVE PIO flight test. The first scale, shown in Fig. 1, illustrates a flow chart where the pilot arrives at a rating by answering a series of questions. During practice runs, pilots occasionally jumped to PIO ratings of 4 when they saw small oscillations that were more of a nuisance than a PIO. These PIORs were given even though Cooper-Harper ratings and pilot comments indicated desired performance and low PIO tendencies. In actual evaluations a second scale, shown in Table 2, was used primarily to help pilots distinguish between "oscillations" and "bobbles". The key phrase to help pilots distinguish between the bobbles and oscillations was "pilot must reduce gain or abandon task to recover."

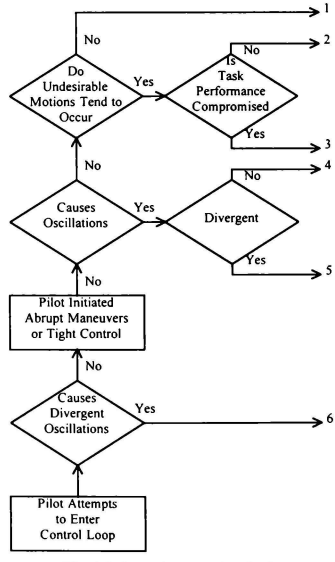


Fig. 1 PIO Tendency Rating Scale

For the landing task, PIO is defined as a sustained oscillation that interfered with the accomplishment of the task and required that pilot reduce his gain or remove himself from the loop. A PIO tendency is defined as an undesirable motion which did not

necessarily interfere with the accomplishment of the task.

Pilots were encouraged to give their comments after each run. That was helpful to the engineers in terms of data analysis.

Table 2 Description of Numerical PIORs

Description	#
No tendency for pilot to induce undesirable motions	1
Undesirable motions tend to occur when pilot initiates abrupt maneuvers or attempts tight control. These motions can be prevented or eliminated by pilot technique.	2
Undesirable motions easily induced when pilot initiates abrupt maneuvers or attempts tight control. These motions can be prevented or eliminated but only at sacrifice to task performance or through considerable pilot attention and effort.	3
Oscillations tend to develop when pilot initiates abrupt maneuvers or attempts tight control. Pilot must reduce gain or abandon task to recover.	4
Divergent oscillations tend to develop when pilot initiates abrupt maneuvers or attempts tight control. Pilot must open loop by releasing or freezing the stick.	5
Disturbance or normal pilot control may cause divergent oscillation. Pilot must open loop by releasing or freezing the stick.	6

Performance Criteria

The following performance criteria were used in the Phase II simulation:

Desired
Touchdown within 5 ft of centerline
Touchdown aim point ±250 ft
Sink Rate ≤ 4 ft/sec
Touchdown airspeed ±5 knots
No PIOs

Adequate
Touchdown within 25 ft of centerline
Touchdown aim point ±500 ft
Sink Rate ≤ 8 ft/sec
Touchdown airspeed +10/-5 knots

These differ from the flight test criteria only in the quantification of the sink rate.

Simulator Descriptions

Ground-based simulators vary in sophistication – from workstations to high-fidelity motion-based simulators to virtual reality environments. Data from two simulation platforms from the Control Integration and Assessment Branch of Wright Laboratory have been used as the basis for this paper.

1. The Large Amplitude Multi-Mode Aerospace Research Simulator (LAMARS) - a 20-ft dome moving-base simulator (used in fixed-base mode).

2. The Mission Simulator-1 (MS-1) - a 40-ft dome fixed-based simulator.

The 18 HAVE PIO configurations were programmed for these simulation platforms. Simulation platforms to be evaluated later are a workstation environment and the LAMARS in motion-based mode.

Mission Simulator-1 (MS-1)

MS-1 is a fixed-based, 40-ft dome with a 180° field of view (FOV) with a high-resolution area of interest (AOI) which has a 40° FOV. The following table lists specifications for the background and AOI:

Table 6 MS-1 Specifications

	Background	Area of Interest
Resolution	11 arc min./line	2.5 arc min./line
Contrast	4:1	7:1
Luminance	0.15 ft-lamberts	0.30 ft-lamberts
Refresh Rate	30 Hz interlaced (60 Hz field rate)	30 Hz interlaced (60 Hz field rate)

Out-the-window scenery was generated with a General Electric Compuscene IVA with two high-resolution channels which can be broken down into six low-resolution channels. It is capable of displaying two million pixels and 4096 polygons. The measured time delay for the PIO simulation using MS-1 was 75ms. The time delay of interest was the delay from when a change occurred in a simulation state variable until that change was observed on the out-the-window display. Delays due to input sampling and the model were not included in this measurement. The Simulation Network Analysis Project (SNAP) tool, designed by engineers at the Control Integration and Assessment Branch, was

used to make these measurements. This tool is capable of collecting and accurately time sampling analog, digital, and ethernet data. In addition it has the unique capability of determining roll and pitch angles of a displayed horizon from a display's video signal. The test consisted of toggling the simulation pitch angle between two discrete values and then measuring the time from when the pitch angle changed until that change was observed in the video signal going to the projectors.

Large Amplitude, Multi-mode Aerospace Research Simulator (LAMARS)

LAMARS is a motion-based, 20-ft dome with two side projectors, each with a 40.5°x30° FOV, and a center projector with a 45°x30° FOV. The total FOV is 135° without gaps. The motion system was not used for this portion of the study. The effects of motion will be examined in the future. The following table gives LAMARS specifications.

Table 7 LAMARS Specifications

Resolution	9 arc min./line horizontal 11 arc min./line vertical
Contrast	17:1
Luminance	0.34 ft-lamberts
Refresh Rate	30 Hz interlaced (60 Hz field rate)

Out-the-window scenery was generated with the same General Electric Compuscene IVA. The time delay for PIO simulation using LAMARS was 90 ms. The difference in time delay from MS-1 was due to the relative synchronization between the simulations and the image generator.

Configuration/Environment/Task Changes

Several changes were made to the simulation in Phase II. The purpose of these changes was to increase the pilot's workload in order to trigger PIOs if any PIO tendencies were present. These changes were: the addition of aerial "gates", additional time delays, the introduction of wind/gust/turbulence and an increase in stick gain.

Gates: Gates were added to the simulation to increase the pilots' workload and to improve visual cues. Fig. 2 shows the runway configuration during the Phase II simulation. The gates were formed by two 210-foot high vertical pylons, each made of three 70-foot, differently colored sections. The first gate was on the glide slope at the initiation point for the lateral-offset correction. and was 150 feet wide. Nominally, this is 3500 feet from the touch-down point. This gate was moved closer to the touchdown point in order to increase the pilot's concentration on the roll task, thus emphasizing the transition to flight path control at the second gate as a possible PIO trigger. The second gate was on the runway centerline, fixed at 1100 feet prior to the touchdown point and was 75 feet wide. Two additional vertical pylons were placed at the touchdown point on each side of the runway. These last pylons were meant to improve the pilots' perception of sink rate at touch-down. Fig. 2 also shows two pairs of horizontal pylons placed 80 feet fore and aft of the touch-down point along the sides of the runway. The horizontal pylons were to help the pilots determine their touch-down dispersion. The large black oval shape on the runway representative of tire marks was also to improve pilot determination of the touch-down point.

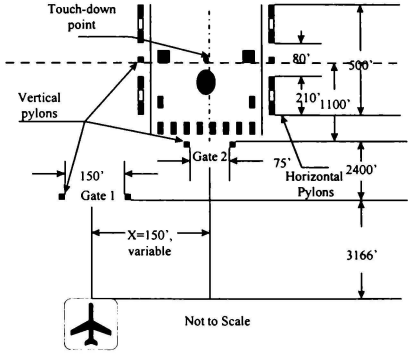


Fig. 2 Runway and Pylons Set-up

Time Delays: Pure time delays were added to the flight control system to add more phase lag to the system. It was hoped that this would counteract the pilots' apparent tolerance for phase lag in the simulator.

Stick Gains: In the flight experiment, a nominal stick gain was chosen by the first pilot who evaluated each configuration. Though this is not the preferred method, it was repeated in the simulation. Increasing stick gains makes the configurations more sensitive to the pilot. This effectively raises the overall system gain,

hopefully countering the pilots' tendency to have lower gains in the simulator.

Wind/Gusts/Turbulence: A wind/gust/turbulence model was implemented in the simulation. This was done to increase the task difficulty and the pilots' workload. A steady quartering wind (45-degree heading) was used with mild-to-medium turbulence, with random medium-to-heavy discrete gusts, and with the nominal gate setting. Random left or right discrete gusts occurred at the first gate (the offset correction) and random up or down discrete gusts occurred at the second gate, which is 40 feet AGL. The discrete gust at the final gate upset the pilot's touchdown solution and forced him back into the loop at high gain.

Pilot Evaluation

The landing task used in the HAVE PIO flight test program was duplicated in the simulator. Topics of the pilot pre-briefing were the tasks, the performance criteria, the runway configuration, Cooper-Harper Rating and PIO Rating scales. Prior to taking evaluation data, each pilot had about twenty minutes for cockpit familiarization for: 1) check out of the simulation for any errors, and 2) exposure to the different characteristics of the various configurations. During the familiarization, the pilots were given their performance data (touchdown dispersions, sink rates, and speeds) after the task was complete. Providing this data to the pilots during the familiarization helped them calibrate where they landed in relation to the touchdown point and their sink rates. These data were not fed back to the pilots during the actual evaluation. The familiarization configurations were not the same as those used in the test matrix. To reduce the test matrix to where all four pilots could finish the test within a reasonable time, only one good and one bad HAVE PIO configurations were evaluated.

A baseline configuration in the Phase II simulation was defined as no wind/gust/ turbulence, nominal stick gain, no additional time delays, and no gates. To avoid any influences from previous configurations, pilots flew the baseline configuration with the nominal gate setting between configuration changes. The following changes were flown singly and not in combination:

1) three different offset gates locations; 150'/2400', 150'/1500', and 300'/2400'. The numerator represents the lateral offset distance from the centerline of the gate to the center of the runway, and the denominator represents the longitudinal distance between the two gates.
2) three different stick gain factors; 2, 2.5, and 3.
3) two pure time delays; two hundred and three hundred milliseconds, and
4) two steady quarterly wind conditions with medium-to-heavy discrete gusts and medium turbulence; twenty-five and forty knots.

Pilots used the long-look technique of Reference [3] for their evaluations. Each configuration was flown a minimum of three times before giving a rating. The three landings consisted of a straight-in approach, a left offset approach, and a right offset approach. Initial conditions were: approach speed of 125 KIAS, glide-slope of two-and-a-half degrees and the altitude of 248 feet above ground level. Lateral offsets were varied for each set-up. Pilots maintained glide slope and heading while flying the offsets by using the first gate and the runway as reference points. Pilots were required to fly through the first gate, then immediately made an offset correction to the centerline of the runway by going through the second gate and attempted to land in the desired area. Pilots were given the freedom to rerun any approaches until they were confident in their rating.

Results

The primary metrics shown here for quantification are the Cooper-Harper Rating and PIO Rating scales. Phase II data analysis represents the effects of the changes to the simulation in terms of pilot ratings. In general, pilot ratings correlated well with pilot performance data, even though outside visual cue limitations in the simulation caused some misjudgment of the landing dispersions and sink rates. Two Head-Up Display (HUD) cues which did help the pilots to determine their landing performance were the flight path marker and the altitude indicator. The vertical and horizontal pylons on the sides of the runway contributed little to the pilots' ability to determine their sink rates.

In order to be considered viable, a change must degrade the flying qualities of only the bad configuration (5-10), and keep the flying qualities of the good configuration (4-1) close to the baseline. A similar trend must exist for PIO ratings.

Three hundred and fifty data runs were collected in Phase II simulation. Fig. 3 shows the average Cooper-Harper Rating of configurations 5-10 and 4-1 versus the changes on MS-1. The horizontal axis of the plot

identifies the baseline configuration and the four general areas of change, with specific values for those changes. This is not meant to imply any linear relationship among these changes. The vertical axis of the plot is the average CHR rating. The dash symbols (configuration 5-10) and the square symbols (configuration 4-1) are for the simulation data, the triangle symbols (configuration 5-10) and the circle symbols (configuration 4-1) are for the HAVE PIO flight test data. Note, flight data exists only for the baseline cases. The vertical dashed lines separate the data into regions of change categories. The first region indicates the average CHR values due to gate changes. The second region indicates the average CHR due to the winds/gusts/turbulence changes. The third region indicates the average CHR values due to the time delay changes. The fourth region indicates the average CHR values due to the stick gain changes. Fig. 3 shows the average simulation CHR of configuration 4-1 matched very well with the flight test data in the baseline configuration. However, the average simulation CHR of the configuration 5-10 was far off from the flight test ratings in the baseline configuration.

Using the usual value of one for the allowable error band, the average CHRs trends due to the changes for both configurations make sense. As expected, when levels of the task difficulty are increased, the pilot workload is also increased for both configurations. The expectations were that as the effective system gain was increased, the bad configuration would show less tolerance for that change, resulting in poor flying quality ratings or PIO. Adding the gates at 150'/2400' and 150'/1500' had little effect on either configuration. However, there were moderate effects on the CHRs of both configurations by reducing the distance between the two gates and increasing their lateral separation (300'/1500'). Data showed moderate effects on the average CHRs of both configurations for winds, gusts, and turbulence at 25 knots and significant effects on the average CHRs of both configurations at 40 knots. Adding 200 milliseconds of time delay to the simulation showed little effect for either configurations. Significant effects were shown for configuration 5-10 with 300 milliseconds time delay. Increasing the stick gain to two times the baseline configuration had little effect on the average CHR of the configuration 5-10, and moderate effects on the average CHR of the configuration 4-1. Increasing the stick gain to two and half times from the baseline configuration had moderate effects on the average CHR of the configuration 4-1. However, it had significant effects on the average CHR of the configuration 5-10. Increasing the stick gain to three times the baseline configuration had very significant effects on the average CHRs of both configurations.

Of the four categories of changes to the simulation, the stick gain was the only change that brought the average CHR of the configuration 5-10 close to the flight test data. However, this change made the average CHR of the configuration 4-1 move away from the flight test value. The changes to the simulation task and environment do not give a clear procedure to improve correlation between flight and fixed-base simulation.

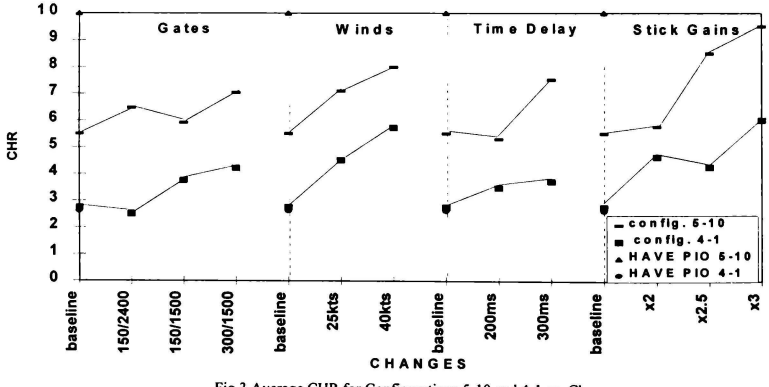


Fig.3 Average CHR for Configurations 5-10 and 4-1 vs. Changes

Fig. 4 shows the average PIORs for configurations 5-10 and 4-1 versus the changes to the simulation. The horizontal axis of the plot is again the baseline and four parameter variations to the simulation. These changes and the symbols are the same as for figure 3. The vertical axis is the average PIORs.

Again, if the allowable error band for PIOR is within one PIO rating, then the average PIORs due to the changes for both configurations are reasonable. As expected, when levels of the task difficulty are increased, the PIORs are also increased for both configurations. Increasing the task difficulty by reducing the distance between two gates or increasing the lateral offset had little effect on the average PIORs of both configurations. Data also showed little effect on the PIORs of both configurations with increasing the magnitude of the winds/gusts/turbulence. Adding 200 milliseconds time delay to the simulation did not affect the PIOR of either configuration. However, adding 300 milliseconds time delay to the simulation had a moderate effect on the PIOR of both configurations. Increasing the stick gains had a significant effect on the PIORs of both configurations with the exception of two times the stick gain which did not affect the PIOR of configuration 5-10. As expected, in general when levels of the task difficulty are increased, PIORs are also increased for both configurations. Among the four changes to the simulation, the stick gain at three times from the baseline configuration was the only change where the PIOR of the configuration 5-10 matched the HAVE PIO flight test data. However, as with CHR, the resulting PIOR for configuration 4-1 did not match the flight test data with this change.

Phase I Time History Analysis

A second area for comparison of the simulation and flight test results was the character of the PIOs which occurred. This of course requires knowing when a PIO happened, which is not always as easy as it seems. FIDO, the in-house analysis program to perform this task, was developed by Dr. Dominick Andrisani while working as summer research faculty from Purdue university. FIDO works on windows of time history data by fitting a sinusoid to the pitch rate and a second sinusoid of the same frequency to the pilot's input.. The fit is done by the MATLAB function "**leastsq**" from the optimization toolbox. FIDO detects PIOs based on the following criteria: phase lag between the pitch rate and the stick input, the least-squares match errors of the pitch rate and of the stick input, and the magnitude of the pitch rate,. The criteria are currently set at: 1) -70 degrees for a phase lag, 2) 0.3 for the pitch-rate match error, 3) 0.6 for the stick-input match error and 4) at least two-degrees-per-second pitch-rate magnitude.

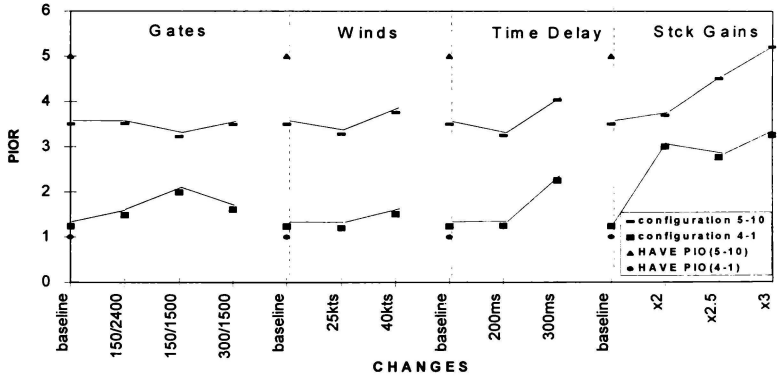


Fig. 4 Average PIOR For Configurations 5-10 and 4-1 vs. Changes

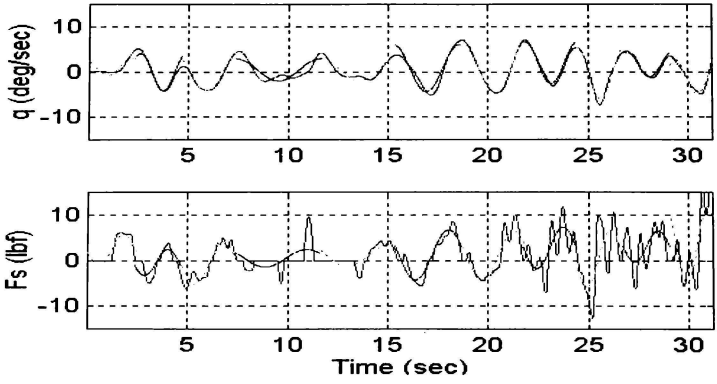


Fig. 5 Pitch Rate and Stick Input Matches

Phase I data is partitioned into sets of data consisting of all runs by a particular pilot with a particular configuration. As mentioned earlier, pilots had at least three chances to see a configuration prior to rating it; one straight-in, one right-offset and one left-offset approach plus any repeats the pilot thought necessary. There are sixty-seven such data sets which were rated four or higher in PIOR, thus at least one time history from each set should indicate the PIO characteristic of that configuration. Pilots were not asked to identify the exact time in which they experienced a PIO. In these sixty-seven sets, engineering analysis found PIO in only forty-nine sets. The current thought is that the sets in which no PIO event was found are examples of the pilot sensing a configuration which might, with more gain, oscillate. The forty-nine data sets, consisting of eighty-six individual runs and one-hundred-and-nine individual PIO events, were used as a database from which to set the criteria in FIDO.

Figure 5 shows a typical time history match of pitch rate in degrees per second and the stick input in pounds. The solid line in the pitch-rate plot indicates the simulation data. The dashed lines are the result of the FIDO match. The current version of FIDO divides the pitch rate data into windows based on two consecutive peaks with a change in magnitude of at least two degrees per second.

Figure 6 shows FIDO output for the example in Figure 5. Each circle represents one FIDO window. A value of zero indicates no PIO in that window. A value of one indicates a possible PIO. There were 11 windows for this run. A PIO is predicted in the seventh window, starting at 18 seconds and ending at 22 seconds. The three values printed at the top of the PIO window are: 1) the phase lag between the pitch rate and the stick input, 2) the amplitude of the matching pitch rate and 3) the PIO frequency.

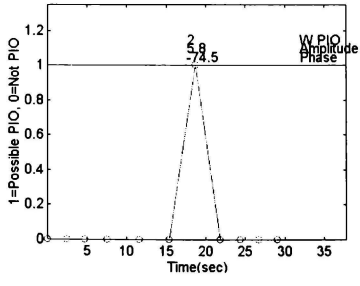


Fig. 6 PIO Events of The Example Run

A comparison of FIDO detections versus engineering analysis of the individual runs gives the following results. There are one-hundred-and-nine PIO events in the database. FIDO found ninety PIO events. Of the ninety FIDO events, eighty-one agreed with engineering analysis and nine did not. Using the current criteria and database, FIDO catches PIOs about seventy-five percent of the time with ten-percent chance of false positives. This is based on a very small set of data. A larger data base will be used to further investigate this result.

Future Plans

LAMARS has undergone projection system upgrades which give it scenery resolution equal to MS-1. A new set of motion gains has been calculated. To understand the effects of the new motion gains on the detection of PIO, at least part of the Phase I and II data will be rerun on LAMARS in the near future.

In April 97, the HAVE LIMIT Test Pilot School Project was conducted at Edwards AFB. The flight test matrix of this project consisted of a few baseline sets of dynamics, several different rate limits and various tracking tasks. These flight test procedures were repeated on the simulator with the same evaluation pilots who flew the flight test. This data exhibits the same trends as the HAVE PIO data discussed previously. Because this data includes performance parameters, it will be used as the truth model for evaluation of rate limiting effects on PIO.

In work parallel to the in-house work at Wright Lab, HAI is looking into ways to increase the pilot gains. These following ideas will be examined on the Wright Lab simulators sometimes in Fall 97:

1) Providing vertical acceleration to the pilots by installing a g-suit,
2) Driving the simulator with a few of the most severe flight PIOs. This will help the engineers a way to qualitatively compare the visual environments between the simulation and flight test. Consider taking one pilot, one case, and having him watch the "video" and then fly the simulated case, back-to-back, and observe the differences in response,
3) Using the Handling Qualities During Tracking task instead of landing task, and
4) Making an attempt to replicate the flight test experiment by implementing a "safety pilot" in the simulation. Of course, the safety trip limits in flight will be modified in the simulation because the pilots' perception in the aircraft response are slightly different between flight and the simulators.

Future plan for FIDO is to expand the data base. We plan to run FIDO with all Phase I simulation data. If the result is the same as it is now, we would like to improve it to a level that FIDO will catch about ninety percent of the possible PIOs with five percent false positives. To achieve this goal, we may have to change the way FIDO matches the data and possibly how to relax the criteria without increasing the false alarms. After improving, we plan to run FIDO with HAVE PIO flight test data. If FIDO continues to show good promise, we will develop FIDO as a real-time PIO detector.

The eighteen configurations of the HAVE PIO flight test program will be implemented on Manned Combat Station for pilots in-the loop simulation sometime this summer. This will complete our "Concepts for Detecting Pilot-Induced Oscillation Using Manned Simulation" program. A technical report will be published in early 98.

Conclusions

As expected, tightening the tasks on the ground-based simulators brings the pilot's C-H and PIO ratings closer to the flight test ratings for the PIO prone configurations. However, the changes also degraded the C-H and PIO ratings of the non-PIO configurations. One observation is that pilots behave much differently on the simulators than they do in flight. It is very difficult to increase the pilot gains by tightening the landing task on the ground-based simulators. Some pilots had a tendency to back out of the task. Therefore the pilots did not fly the airplane as a closed-loop system. When the pilots fly a high-gain task at a low gain, they may not see the PIO characteristics of the configurations. If they do, the PIO won't be as severe as they fly the configurations at high gains. Looking for PIOs on the ground-based simulator may require certain pilot skills and some particular tasks where the pilots can not back out. As stated earlier, the ground-based simulators can duplicate the flight test results. However, we do not yet know how to make ground-based simulators successful at detecting PIO problems prior to flight test.

References

1. Bjorkman, E. A., "Flight Test Evaluation of Techniques to Predict Longitudinal Pilot Induced Oscillations", AIAA Paper, 1986.

2. Kish, Leggett, Nguyen, Cord, and Slutz, "Concepts For Detecting Pilot-Induced Oscillation Using Manned Simulation" , AIAA Paper, 1996.

3. Dukes, T. A., Guideline for Designing Flying Qualities Experiments, Report No. NADC-85130-60, June 1985.

AIAA-97-3731

A COMPARISON OF REAL-TIME SOLUTIONS TO ALL-ATTITUDE ANGLES OF AN AIRCRAFT

Sungsik Shin*, Sugjoon Yoon**
Korea Institute of Aerospace Technology, Korean Air
Taejeon, Korea

Abstract

Various methods available in the literature are compared experimentally to find the most suitable real-time solution to all-attiude angles of an aircraft. Among them are methods of quaternion, dual Euler, and direction cosine.

Quaternion and directon cosine methods have been widely used in the field to avoid singularity even if it introduces non-physical parameters causing difficulties when physical constraints should be applied to rotational state variables. This is why conventional Euler method is still preferred in numerical applicatons without singularities. Dual Euler has advantages over quaternion and direction cosine methods in the sense, which takes advantage of alternate coordinate transformations in order to avoid rotational singularities.

In the paper these methods are numerically compared by a non-aerodynamic 6 DOF rigid model. Dual Euler method turns out to be superior to the others in the applications because it shows better numerical accuracy, stability, and robustness in integration step sizes. Dual Euler is affordably less efficient than quaternions in computational cost. Numerical accuracy and stability, which allow larger integration step sizes, are more critical in modern real-time applications than computational efficiency because of today's increased computational power. If quaternions are required because of constraints in computation time, then suppression mechanism should be provided for algebraic constraint errors which will eventually add computational burden. The same logic applies to the direction cosine method.

Introduction

Euler angles[1] and quarternions[2,3] have been used traditionally for typical flight simulations in the simulation community. Euler angles have their own merits in simple mathematical expression and physical interpretation of the parameters. But they suffer from singularities at vertical flights. Thus Euler angles are limited to simulation of commercial aircrafts. In order to overcome this singularity problem quaternions are applied to fighter simulations, since vertical or reversed flights are quite common in fighter aircrafts. The quaternion method has its own limits which include more complex expression of algebraic differential equations and difficulties in physical interpretation of the parameters.

In the literature exist there some other methods[4,5], for solving all-attitude angles of an aircraft, claimed to be superior to traditional Euler angles and quaternions. Dual Euler[5] method and direction cosine method are among those. Dual Euler divides all-attitude angles into two zones, and applies two alternate sets of Euler angles to avoid singularities in relavant zones. The core concept of dual Euler is to take advantage of different sets of Euler angles and switch to a non-singular set when rotational singularity points are crossed over when using a set of Euler angles. Direction cosine[2] method avoids the singularity problem by integrating direction cosine rates and solving rotational kinematics equations resulting from orthogonality conditions of the direction cosine matrix. Direction cosine method is similar to quaternions in a sense that it is difficult to apply physical constraints to the parameters when necessary.

* Research Engineer
** Head of Simulation Group / Member AIAA

Copyright © 1997 by the
American Institute of Aeronautics and Astronautics, Inc.
All rights reserved

In the paper three solutions such as quaternions, dual Euler, and direction cosine are compared experimentally to identify the most appropriate one in real-time application. Especially, real-time aspects are concentrated in the comparison, which are numerical accuracy, stability, and computational load. Test simulations say that the dual Euler method is superior to the others in accuracy and stability. With small integration step sizes are dual Euler and direction cosine methods more accuarate and stable than quaternions. Dual Euler is more robust in step sizes than quaternion and direction cosine methods. Dual Euler is affordably less efficient than the other methods in computational burden. However, its additional computational cost can be considered acceptable considering overall amount of simulation codes in modern applications and recent growth in computation capability. If quaternions are required because of constraints in computation time, then suppression mechanism should be provided for algebraic constraint errors which will eventually add computational burden.

Numerical Solutions to All-Attitude Angles

In this comparison is traditional Euler method excluded because it comprises intrinsic singularity and cannot be applicable to all-attitude angles. Three numerical solutions available in the literature are introduced, which are quaternion, dual Euler, and direction cosine methods. Their natures are briefly explained here.

Quaternion Method

Coordinate transformation from aircrafts' rotating body axes (x_0, y_0, z_0) to fixed inertial axes (x, y, z) is represented by Eqn. (1), when quaternions (e_0, e_1, e_2, e_3) are adopted rather than Euler angles.

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} e_0^2+e_1^2-e_2^2-e_3^2 & 2(e_1e_2+e_0e_3) & 2(e_1e_3-e_0e_2) \\ 2(e_1e_2-e_0e_3) & e_0^2-e_1^2+e_2^2-e_3^2 & 2(e_2e_3+e_0e_1) \\ 2(e_0e_2+e_1e_3) & 2(e_2e_3-e_0e_1) & e_0^2-e_1^2-e_2^2+e_3^2 \end{bmatrix} \begin{bmatrix} x_0 \\ y_0 \\ z_0 \end{bmatrix} \tag{1}$$

Here quaternions are governed by the equations as follows:

$$\begin{aligned} \dot{e}_0 &= -\frac{1}{2}(e_1P + e_2Q + e_3R) + k\lambda e_0 \\ \dot{e}_1 &= \frac{1}{2}(e_0P + e_2R - e_3Q) + k\lambda e_1 \\ \dot{e}_2 &= \frac{1}{2}(e_0P + e_2R - e_3Q) + k\lambda e_1 \\ \dot{e}_3 &= \frac{1}{2}(e_0R + e_1Q - e_2P) + k\lambda e_3 \end{aligned} \tag{2}$$

where P, Q, and R are roll, pitch, and yaw rate respectively. Integration step size, h, must satisfy the condition of $kh \leq 1$ for the simulation not to diverge. k is a constant, and determined experimentally for appropriate simulation result to be achieved.

There applies a constraint equation to quaternion parameters:

$$\lambda = 1 - (e_0^2 + e_1^2 + e_2^2 + e_3^2) \tag{3}$$

Thus the quaternion approach comprises 4 1st-order differential kinematic equations and 1 algebraic constraint equation for the problem of 3 degrees of rotational freedom.

Direction Cosine Method

The direction cosine matrix defines the transformation between inertial and body axes. Each direction cosine represents the projection of a unit vector in one axis system onto a unit vector in the other axis system. The direction cosine method is to take advantage of the relation:

$$[\dot{a}_{ij}] = \begin{bmatrix} 0 & R & -Q \\ -R & 0 & P \\ Q & -P & 0 \end{bmatrix} [a_{ij}] \tag{4}$$

The 9 Direction cosine rates $[\dot{a}_{ij}]$ in Eqn. (4) are integrated, and obey the orthogonality conditions leading to the following constraint equations:

$$\begin{aligned} a_{11} &= a_{22}\, a_{33} - a_{23}\, a_{32} \\ a_{21} &= a_{13}\, a_{32} - a_{12}\, a_{33} \end{aligned}$$

$$
\begin{aligned}
a_{31} &= a_{12}\,a_{23} - a_{13}\,a_{22} \\
a_{12} &= a_{23}\,a_{31} - a_{21}\,a_{33} \\
a_{22} &= a_{11}\,a_{33} - a_{13}\,a_{31} \\
a_{32} &= a_{13}\,a_{21} - a_{11}\,a_{23} \\
a_{13} &= a_{21}\,a_{32} - a_{22}\,a_{31} \\
a_{23} &= a_{12}\,a_{31} - a_{11}\,a_{32} \\
a_{33} &= a_{11}\,a_{22} - a_{12}\,a_{21}
\end{aligned} \tag{5}
$$

These equations result in the states of direction cosines at the present integration step, and the state values are used for integration of Eqn. (4) at the next step. That is, the singularity problem is resolved by integrating 9 linear differential equations for rotational kinematics rather than 3.

Dual Euler Method

Dual Euler applies two different sets of Euler angles to avoid singularities by switching from one to the other. Two sets, ordinary and reversed, are related by Eqn. (6) for the direction cosine.

$$
\begin{aligned}
A &= [\Phi]_x\,[\Theta]_y\,[\Psi]_z \\
&= [\Theta_r]_y\,[\Phi_r]_x\,[\Psi_r]_z
\end{aligned} \tag{6}
$$

An ordinary set of Euler equations shown in Eqn. (7) are integrated until θ comes close to vertical points, $\theta = \pm\pi/2$.

$$
\begin{aligned}
\dot{\phi} &= P + \dot{\psi}\sin\theta \\
\dot{\theta} &= Q\cos\phi - R\sin\phi \\
\dot{\psi} &= (Q\sin\phi + R\cos\phi)/\cos\theta
\end{aligned} \tag{7}
$$

where ϕ, ψ, and θ are roll, yaw, and pitch angle respectively.

In the vicinity of the singular points the set of Euler equations switches to a different set of reversed Euler equations defined in Eqn. (8).

$$
\begin{aligned}
\dot{\phi}_r &= P\cos\theta_r + R\sin\theta_r, \\
\dot{\theta}_r &= Q - \dot{\psi}_r\sin\phi_r, \\
\dot{\psi}_r &= (-P\sin\theta_r + R\cos\theta_r)/\cos\phi_r,
\end{aligned} \tag{8}
$$

Here ϕ_r, ψ_r, and θ_r are reversed roll, yaw, and pitch angle respectively. As can be seen, this set has also singularities at $\phi_r = \pm\pi/2$. When the singular points of ϕ_r come closer during numerical simulation, the ordinary set of Euler equations is recovered. Here $\cos\phi_r = 0$ is equaivalent to $\theta = 0$ by the relationship in Eqn. (6). The switch-over criteria are typically recommended to be $\theta = \pm\pi/4, \pm 3\pi/4$.

Numerical Tests

The goal of numerical tests is to identify which algorithm among quaternion, dual Euler, and direction cosine methods is the most appropriate in the real-time environment. The criteria are accuracy, stability, computational load, and robustness in integration step sizes. The three methods are applied to a non-aerodynamic 6 DOF rigid-body model[5], and compared at various conditions such as different integration algorithms and step sizes. The terminology of non-aerodynamic means that 6 DOF rigid body is driven by Eqn. (9), not by aerodynamics and other forces.

$$
\begin{bmatrix} P \\ Q \\ R \end{bmatrix} = \begin{bmatrix} \pi/4\sin\pi t \\ \pi \\ -P \end{bmatrix} \tag{9}
$$

Since it is impossible to obtain an analytic solution to the problem, Runge-Kutta 4th-order integration is used to yield a solution assumed to be exact. Errors in rotational state variables are monitored in the paper only when Adams-Bashforth 2nd-order integration is applied with step sizes h = 0.05 sec and 0.01 sec. However, it is observed that other integration methods result in similar trends.

It is not a trivial task to determine appropriate value of k in the quaternion method. In the tests shown here k = 10.0 is selected for best performance after scores of test runs. Fig. 1 through Fig. 3 show errors of three methods in the rotational variables when h = 0.01 sec is adopted. As can be seen in the plots, both dual Euler and direction cosine methods are acceptably accurate, and stable compared with quaternions.

The quaternion method has a problem of algebraic-differential equations[6], and the divergence of numerical errors can be expected. Fig. 4 through Fig. 6 are regarding a larger step size, h = 0.05 sec. The plots show that the dual-Euler method is more robust in step sizes than the direction cosine method. The direction cosine method yields unacceptable errors in pitch angle about t = 6.0 sec, which is equivalent to $\theta = -\pi/2$. Direction cosine fails to compute the value of θ by the relation[1]:

$$\theta = \sin^{-1}(-a_{13}) \qquad (10)$$

Because of truncation errors, the absolute value of the argument in Eqn. 10 at the time becomes larger than 1 when AB-2 and t = 0.05 are combined. But RK-4 does not violate the limit even when t = 0.05 is used. The integration tool, Matlab, automatically assigns 0 if it fails to compute $\sin^{-1}$. That is why we have unacceptable errors around t = 6 sec. If the simulation lasts for longer time or larger step sizes are used, which means larger truncation errors, then the direction cosine method inevitably reveals similar failures more often.

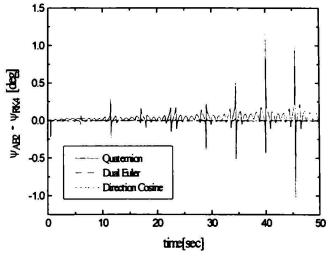


Fig. 2. Comparison of numerical errors in yaw angle ψ. (h=0.01, AB-2).

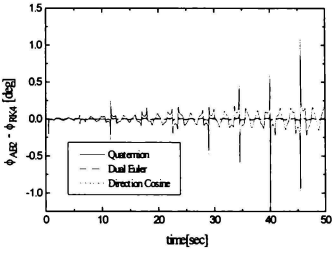


Fig. 3. Comparison of numerical errors in roll angle ϕ. (h=0.01, AB-2).

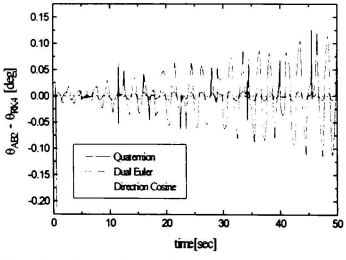


Fig. 1. Comparison of numerical errors in pitch angle θ. (h=0.01, AB-2).

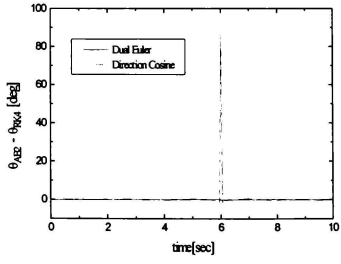


Fig. 4. Comparison of numerical errors in pitch angle θ. (h=0.05, AB-2).

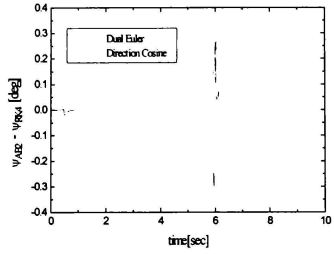


Fig. 5. Comparison of numerical errors in yaw angle ψ. (h=0.05, AB-2).

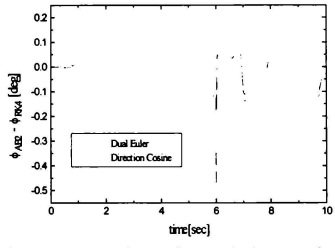


Fig. 6. Comparison of numerical errors in roll angle ϕ. (h=0.05, AB-2).

With small integration step sizes are dual Euler and direction cosine methods more accuarate and stable than quaternions. Dual Euler is more robust in step sizes than quaternion and direction cosine methods. The reason is that only the dual Euler method is free from algebraic constraint equations among the three methods. The constraints are constantly disturbed by truncation errors during integrations, and there is no mechanism applied in the original methods to suppress the divergence of errors.

Computational burdens of the three methods are compared at Table 1. The computer used in the test is a PC 486 system with 66MHz clock speed and 16 Mbyte RAM. AB-2 and RK-4 are combined with a step size, h = 0.01 sec in the comparison. Relative time consumptions of the others to the quaternion method's are presented in the table. The result says that the dual Euler method is the least efficient among the three, but the differences are negligible considering today's rich computational power and additional computational load to integration of state variables in whole flight simulation.

	Quaternion	Dual Euler	Direction Cosine
AB-2	1	1.22	1.11
RK- 4	1.82	2.55	2.03

Table 1. Relative time consumptions of solutions to all-attitude angles (h=0.01 sec)

Some additional test runs are performed to identify at what pitch angles switch-overs of Euler angles should be made for the best result in the dual Euler method. As mentioned earlier in this paper, recommended switch-over pitch angles are $\theta = \pm\pi/4, \pm 3\pi/4$. The test simulations show that the values of switch-over angles do not cause much effects on the accuracy of simulations, if the angles are not close enough to the singular points.

Conclusions

Dual Euler is affordably less efficient than the other methods in computational burden. However, its additional computational cost can be considered acceptable considering overall amount of simulation codes in modern applications and recent growth in computation capability.

In conclusion, dual Euler method is superior to the others in accuracy and stability, robustness in integration step sizes. If quaternions are required because of constraints in computation time, then suppression mechanism should be provided for algebraic constraint errors which will eventually add computational burden. The same logic applies to the direction cosine method.

References

[1]Rolfe J. and Staples K., Flight Simulation, Cambridge Univ. Press, 1986.

[2]Cardullo F., "Mathematical Modeling", Lecture Notes of 1994' Flight Simulation Update, State Univ. of New York, Binghamton, January, 1994.

[3]Robinson A., "On the Use of Quaternions in Simulation of Rigid-Body Motion", WADC TR 58-17, December, 1958.

[4]Mebius J., "Mathematics for Simulation of Near-Vertical Aircraft Attitudes", Proceedings of Make it Real, CEAS Symposium on Simulation Technology, October, 1995.

[5]Huang X., "A Dual-Euler Method for Solving All-Attitude Angles of the Aircraft", Proceedings of 1993' AIAA Flight Simulation Technologies Conference, pp. 257-262, August, 1993.

[6]Yoon S., "Real-time Simulation of Constrained Dynamic Systems", Ph.D. Dissertation, Univ. of Michigan, Ann Arbor, 1990.

AIAA-97-3786

DYNAMIC INVERSION APPLIED TO THE F-117A

Richard Colgren, Ph. D.
Lockheed Martin Skunk Works
1011 Lockheed Way, Bldg. 608, Plant 10
Palmdale, CA 93599-2525

and

Dale Enns, Ph. D.
Honeywell Technology Center
3660 Technology Drive, Minneapolis, MN 55418

ABSTRACT

The objective of the work documented in this paper was to develop design procedures for applying multivariable control theory to fighter aircraft control law design. Specifically, the dynamic inversion multivariable control design method was applied to the F-117A to meet stability, performance, and disturbance rejection requirements. Precompensation uses pilot inputs and creates commands for the feedback loop. Three control variables were constructed from various sensed quantities. These were shaped to generate the desired values for the derivatives of the controlled variables.

INTRODUCTION

This paper documents design work accomplished for the US Air Force, Wright Laboratories - Flight Dynamics Directorate, Dayton, Ohio under contract number F33615-92-C-3607 "Application of Multivariable Control Theory to Aircraft Control Laws". This work was conducted by companies of the Lockheed Martin Corporation reporting to the Honeywell Technology Center. The objective of this contract was to develop design guidelines for application of multivariable control theory to aircraft control laws. This study was part of a three-year U.S. Air Force contract [WL96]. It was an entirely analytic study. Flight control systems were developed for three different aircraft (F-117A, F-16, and YF-22). In this study four different maneuvers using three different multivariable control design methods (μ-synthesis, dynamic inversion, and eigenstructure synthesis) were evaluated. Full nonlinear databases were used to satisfy nonlinear simulation requirements. Specifically covered in this paper are the dynamic inversion multivariable control designs developed by the Lockheed Martin Skunk Works for the F-117A aircraft (see Figure 1). This design work included system performance validation using linear and nonlinear models and non-real time nonlinear simulation.

The basic advantage claimed for Multivariable Control Theory (MCT) methods is that multiple control loops are closed simultaneously, meeting a performance criterion. When using classical design methods (as originally used in developing the F-117A flight control system) successive single loop closures are performed, whereby the desirable characteristics achieved in the first closure may be lost in the second. This advantage is very important with the advent of control-configured vehicles and strong cross-coupling between vehicle axes [Do84].

Results indicate that methods studied under the MCT effort can be satisfactorily applied to real aircraft control problems. Dynamic inversion controllers have proved especially portable between aircraft types.

Copyright © 1996 by Richard Colgren & Dale Enns. Published by the American Institute of Aeronautics and Astronautics, Inc. with permission.

F-117A DESCRIPTION

The F-117A shown in plan and side view in Figure 1 is a single seat fighter-bomber in the 36,000 to 52,000 pound class. All weaponry is stowed in internal bays, which open under the center area of the fuselage. Primary pilot controls are a center stick, pedals and power levers. The combination of shape and attainable C.G. location results in an aircraft which is unstable in pitch and yaw for much of its flight envelope. It exhibits a large dihedral effect and a nose up pitching moment with sideslip and high angles of attack. It is beyond the human pilot's capability to compensate for these characteristics. The analog fly-by-wire flight control system (FCS) on the F-117A trims the aircraft and constrains its flight within a safe envelope. The FCS provides stability as defined in MIL-F-8785C for controlled flight [Mi80]. There is no mechanical backup.

When the F-117A program was started, the only fly-by-wire system in production was on the F-16, which had just entered operational service. To reduce technical risk for the F-117A program and avoid the cost of new FCS component development, maximum use was made of mature F-16 technology.

The horizontal control surfaces are the four elevons on the sharply swept wings. They are labeled left outboard, left inboard, right inboard, and right outboard. There is no horizontal tail. Vertical control surfaces are the fully rotating left and right fins of the V-tail. The fins consist of approximately 70% of the V-tail and pivot on a shaft extending through the stub fins. Surface rate and position limits are given in Table 1. The grids on either side of the

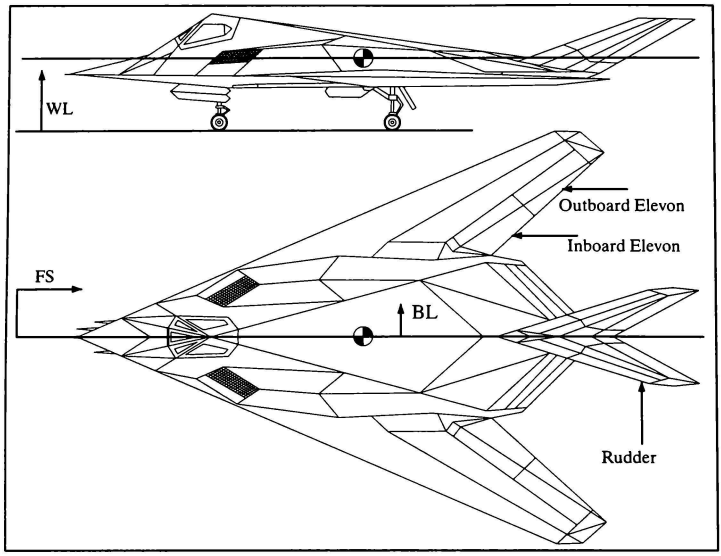


Figure 1: F−117A Plan and Side Views

canopy are the inlets for the two GE F404 engines. Exhaust nozzles are the rectangular openings parallel to the trailing edge of the fuselage.

Table 1: Surface Rates and Position Limits

Control Surface	Rate (deg/sec)	Position (deg)
Inboard Elevon	± 150	± 45
Outboard Elevon	± 200	± 45
Fins	± 84	± 19

DYNAMIC INVERSION

The dynamic inversion method used in this work is documented in [Ho92 and WL96]. The variables LCV, MCV, and NCV represent the variables to be controlled in the roll, pitch, and yaw axes respectively. They are a blend of various sensed signals. A five step process is then used to implement a dynamic inversion controller.

Step 1: Choose a blend of variables to construct the controlled variables. In a manned aircraft these are chosen to provide adequate handling qualities. For example, the choice of MCV must reflect the pilot's control desires. Pure n_z was initially used to emulate the g command nature of the classically designed F−117A control laws. Later a C^* MCV was used to provide a natural change in control priority from rotation to translation as airspeed increases. Note that the open loop responses of the controlled variables must have no right half plane zeros.

Step 2: The aircraft model is defined. The key component in this step is the development of a simplified aerodynamic model that can be run in real time. This least squares model can be written as $C_k = C_k(\alpha, M)\,\eta$. Here C_k is the vector of least squares aerodynamic coefficients, C_k is the vector of aerodynamic coefficients in the complete aerodynamic database and is a nonlinear function of angle of attack (α) and Mach number (M). Finally, $\eta = [1, \beta, bp/2V, \bar{c}q/2V, br/2V, \delta_a, \delta_e, \delta_r]^T$ is the vector of independent variables. These variables are the sideslip angle, the scaled roll, pitch, and yaw rates, and the roll, pitch, and yaw control surface deflections.

Step 3: Next the desired characteristics of the controlled variable ($\dot{V}^{des}$) are specified. Ideally, the dynamic inversion portion of the control law, along with open loop aircraft dynamics, produces an integration. The desired dynamics are $\dot{V}^{des} = b\,[f_c V_{cmd} - V + f_i z]$, where $z = b(V_{cmd} - V)$. The two standard cases are

proportional feedback ($f_c=1$, $f_i=0$) and proportional plus integral feedback ($f_c=0.5$, $f_i=0.25$).

Step 4: Here forces and moments corresponding to undesired aerodynamic, gravitational, and/or inertial contributions are modeled as functions of the measured state variables. Opposing forces and moments are then commanded to negate the undesired contributions. In the simple case of the linearized aircraft roll axis model the dynamic equation is $\dot{p} = L_p p + L_{\delta_a}\delta_a$. Inverting this dynamic equation for the control input yields $\delta_a = \frac{1}{L_{\delta_a}}[\dot{p} - L_p p]$. In the general case, the nonlinear equations of motion are $\dot{x} = F(u,x) \approx f(x) + g(x)u$. Here x is the vector of state variables and u is the vector of control surfaces. The time derivative of the vector of the control variables is $\dot{V} = \frac{\delta V}{\delta x}\dot{x} = \frac{\delta V}{\delta x} f(x) + \frac{\delta V}{\delta x} g(x)u$ $= a(x) + b(x)u$. The dynamic inversion part of the control law is written in terms of the desired control variable as $u^{cmd} = \left[\frac{\delta V}{\delta x} g(x)\right]^{-1}\left[\dot{V}^{des} - \frac{\delta V}{\delta x} f(x)\right]$ $= b(x)^{-1}\left[\dot{V}^{des} - a(x)\right]$.

Step 5: In the final step the handling qualities are tuned using precompensation. For example, stick and rudder pedal gains and shapes are defined.

NONLINEAR SIMULATION

This phase used the nonlinear F−117A Fortran simulation including nonlinear propulsion and aerodynamic databases to directly compare the controllers developed using the multivariable control methods with each other and with the classically designed F−117A control laws. The classical F−117A control laws are gain scheduled to accommodate parameter variation throughout the flight envelope. Dynamic inversion control laws do not require gain scheduling but instead use the least squares aerodynamic database. The aerodynamic and propulsion models vary in a nonlinear manner throughout the simulated maneuver.

Two manual and two automatic maneuvers are used to validate the control system design. All of the maneuvers are "unconventional" in the sense that they require control of dynamic coupling between axes. The manual maneuvers consist of a rapid pull into the angle of attack limiter followed by a bank angle capture while remaining on the limiter for: 1) a clean configuration and 2) simulated combinations of rudder and elevon failures. The automatic modes consist of: 1) a coupled glide slope and heading capture and 2) a Pilot Activated Automatic Recovery System (PAARS) maneuver.

The rapid pull−up is initiated at 300 knots equivalent airspeed, 1g, wings level, 10,000 ft, and at a power setting for level flight. The engine power lever is advanced to the maximum power setting (military power) as the maneuver is initiated. Roll stick initiation is varied to find the "worst case" coupling. Full stick inputs are made into both axes with the intent of capturing an eighty degree bank angle while remaining on the angle of attack limiter. The roll input is a combination of the nonlinear pilot input and a subsequent linear model for the terminal bank angle capture task. Operationally, this maneuver might be initiated as a defensive break turn or to reduce an opponent's missile launch envelope. This maneuver is designed to provide a difficult control integration task for the F−117A.

The second manual maneuver utilizes the ability of modern control methods to handle the integration of multiple surfaces. The F−117A suite of control surfaces and their coupling characteristics are such that SISO (Single Input Single Output) classical methods are more than adequate for control synthesis. In order to provide an operationally applicable scenario and increase the need to treat surface integration as a MIMO (Multiple Input Multiple Output) task, the second manual maneuver is performed on various failed states of the surfaces. The failures are modeled by artificially constraining combinations of rudders and elevons to fixed neutral deflections.

The landing task provides a design challenge since lower airspeeds require higher feedback gains and large amplitude control deflections for maneuvering. Often the takeoff and landing phases size the control power for the vehicle. MIL−F−8785C provides automatic flight control mode requirements which can be utilized in a analytical design and validation task. The nonlinear simulation includes pitch and roll attitude control, heading control, and altitude control.

The second automatic maneuver utilizes the Pilot Activated Automatic Recovery System (PAARS) designed for the F−117A. This automatic maneuver exercises the control design in high pitch and roll attitude situations through large ranges of coupled attitude changes.

PAARS provides an all−attitude recovery capability to protect against pilot disorientation due to sensor/display failure, pilot vertigo induced by the lack of external visual cues at night, and/or pilot distraction due to high workload. PAARS generates commands to the Flight Management System (outer loop/autopilot) which in turn generates commands to the Flight Control System (inner loop). The recovery commands that PAARS generates emulate the commands from a fully lucid pilot in the same flight situation. PAARS utilizes a sector scheme which divides the attitude envelope into seven operating regions. This scheme is shown in Figure 2.

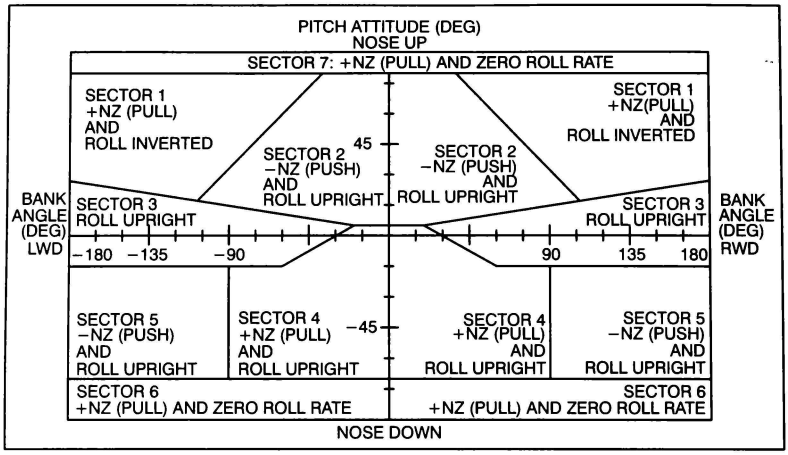


Figure 2: PAARS Recovery Versus Attitudes

PILOT MODEL

The F−117A simulation contains a simple nonlinear pilot model. In the pitch axis (see Figure 3) there are n_z or angle of attack (α) command models which use a lead−lag followed by a Proportional and Integrator (PI) filter to generate zero steady state error. The roll axis model is more complex, with different modes depending on whether a full stick command is required (see Figure 4). For small bank angle errors (and lower roll rates) a bank angle is held with damping provided by roll rate feedback. When the Δ command is large it goes to a full stick input, computing the time to ease back on the stick command to acquire the desired bank angle. In the yaw axis during approach to a landing the pilot model achieves a sideslip or a constant lateral−directional flight path (i.e. the pilot model controls the aircraft to stay aligned on the centerline of the runway, see Figure 5). This model is used with the roll axis pilot model. The power lever is commanded open loop (i.e. in the acceleration portion of the pull and roll maneuver).

SIMULATION RESULTS

The first maneuver simulated is a PLA (power lever angle) increase with a pull and roll to 80 degrees bank angle. This response is shown in Figure 6. Some work is required to make the pitch response acceptable to a pilot. The g response is too sluggish during the initial pull up. Rolling at high angle of attack causes α to increase further. Elevon deflections are too large and rate limited during the roll maneuver (deflections should be <20 degrees due to hinge moment limits). On the F−117A the inboard elevon no load rate is 120 degrees/second while the outboard elevon no load rate is 160 degrees/second. Fin deflections saturate and rate limit during the roll (fin deflection should be <14 degrees). The actual maximum no load rate is 70 degrees/second while at this condition it is approximately 55 degrees/second.

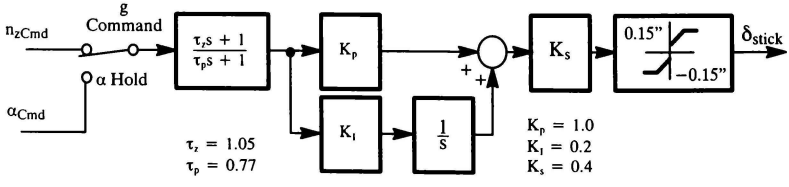
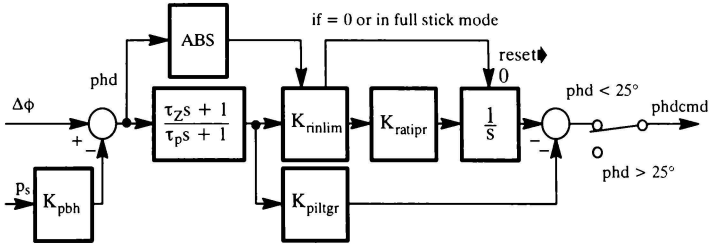
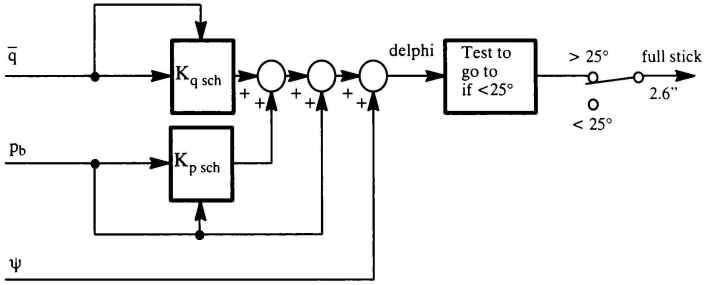


Figure 3: Longitudinal Pilot Model

Partial stick mode



Full stick mode: on if phd>25, off if delphi+ψ>phicmd(right) or <phicmd(left)



Roll stick command

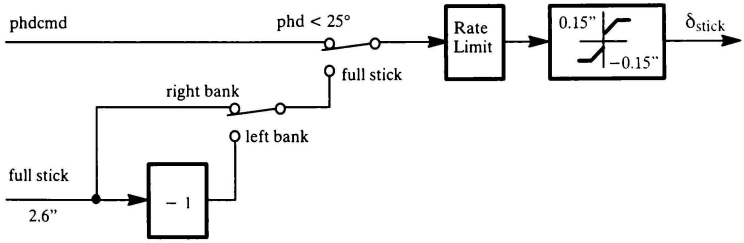


Figure 4: Lateral Pilot Model

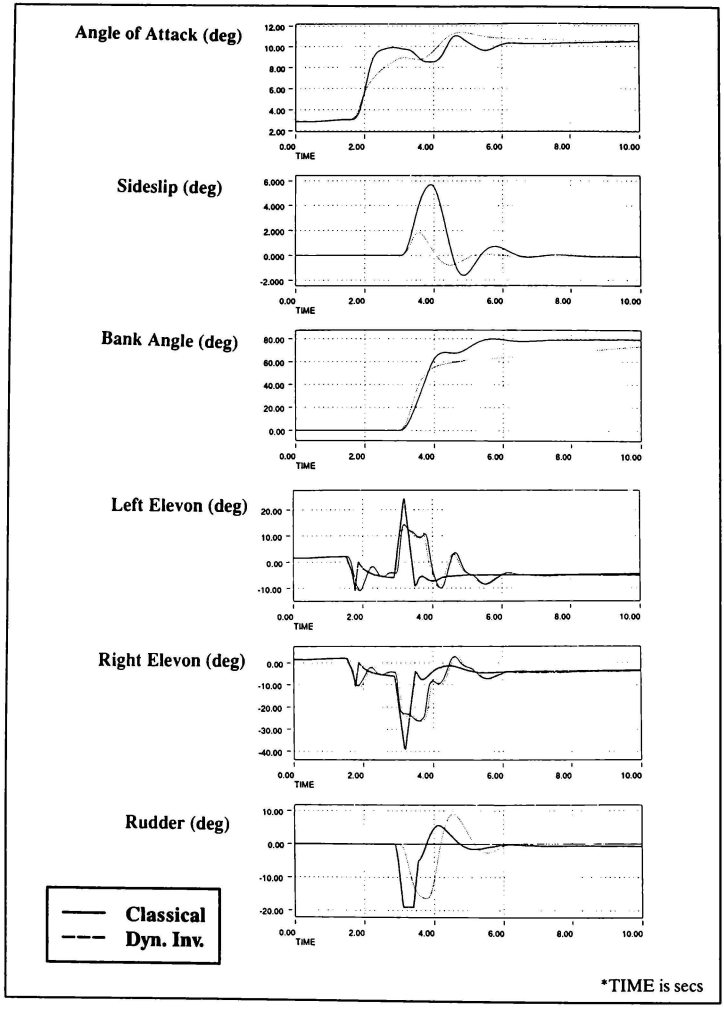


Figure 7: F−117A Roll and Pull Maneuver with Failure

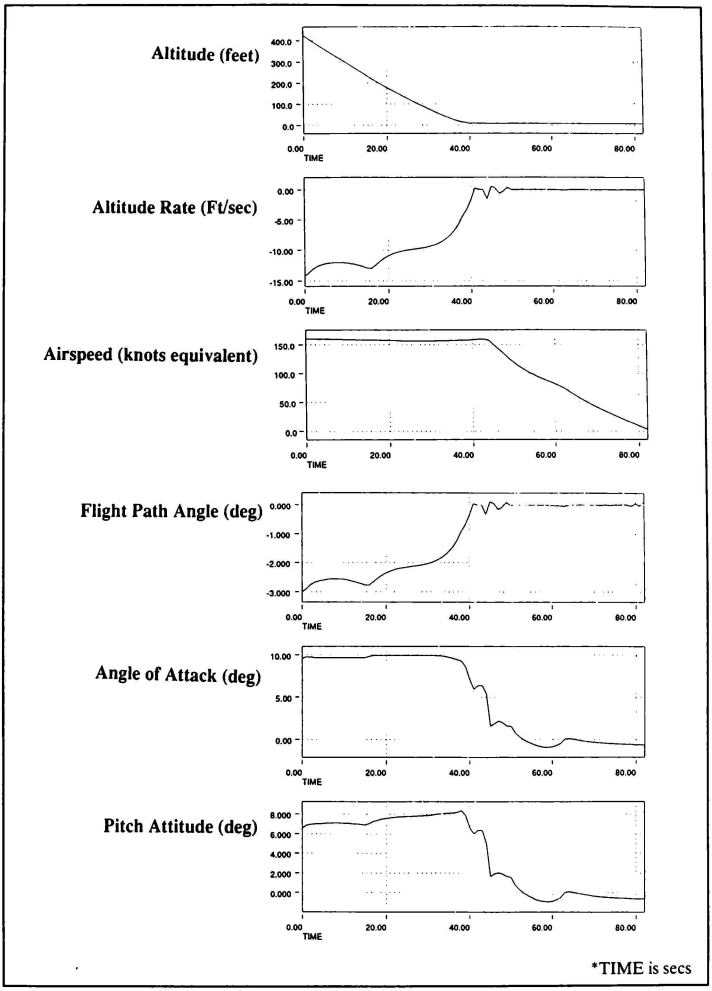


Figure 8: F−117A Approach and Landing

CONCLUSIONS AND SUMMARY

The largest integration issues in applying the dynamic inversion methodology to the F−117A were not related to the control laws. Instead, they were software related issues (i.e. common blocks and call lists, single versus double precision, trimming the control laws after completing aerodynamic trim). A small least squares aerodynamic database proved sufficient for use in the control laws for the inversion process. An excellent fit to the nonlinear database was obtained using linear trim points. Few points were required to define the aerodynamic curves for this database.

The control surface position and rate limits in the dynamic inversion control laws were not constrained as in the existing F−117A control laws, but were simplified to fixed values. Ground effects proved to be much weaker than expected in the latest F−117A aerodynamic database. This was out of the scope of this effort but needs to be explored further.

The dynamic inversion control design methodology was successfully applied to the F−117A. Full nonlinear simulations were used to compare closed loop dynamic responses with those generated using the existing F−117A control laws. Multiple maneuvers were examined to demonstrate performance at many points in the flight envelope without the gain scheduling required by the existing classically developed F−117A control laws.

REFERENCES

[Do84] Doyle, J. C., et al., **"Lecture Notes, ONR/Honeywell Workshop on Advances in Multivariable Control,"** Honeywell Corp, Minneapolis, MN, 1984.

[Ho92] "Dynamic Inversion Based Flight Control Laws for the HARV," Honeywell Systems and Research Center, Minneapolis, MN, 3 March 1992.

[Mi80] "Military Specification, Flying Qualities of Piloted Airplanes," MIL−F−8785C, 5 November 1980.

[WL96] "Application of Multivariable Control Theory to Aircraft Control Laws − Final Report: Multivariable Control Design Guidelines," WL−TR−96−3099, Flight Dynamics Directorate, Wright Laboratory, OH, May 1996.

SIMULATION STUDY OF VISTA/F-16 MANEUVERABILITY ENHANCEMENT USING FOREBODY VORTEX CONTROL

Phillip D. McKeehen* and Thomas J. Cord#
Wright Laboratory(WL/FIGC)
Wright Patterson Air Force Base, Ohio 45433-7531

ABSTRACT

A piloted experimental study of potential enhanced task performance resulting from improved high angle-of-attack aerodynamic and flight control capability was conducted in the Air Force Research Laboratory's engineering flight simulator facility. The simulation database used was representative of the aerodynamics and inertias of the Variable-stability In-flight Simulator Test Aircraft (VISTA) / F-16. The VISTA variable-stability control laws were not used. Three flight test pilots evaluated both baseline and three modified versions of the simulated aircraft using a variety of high angle-of-attack tasks. Aerodynamic modifications were based on wind tunnel data from a previous effort which examined various means of extending the aircraft angle-of-attack limits. These focused primarily on the lateral-directional characteristics in the twenty-nine to thirty-seven degree range. Flight control modifications came from a new approach to control of lateral-directional dynamics which used variable structure control and describing functions. This controlled the forebody vortices to achieve improved roll coordination. This paper presents the results of analyzing the entire set of experimental output data for the effects of the configuration changes on high angle-of-attack maneuverability and departure resistance. The results show that use of the modifications greatly increases departure resistance and provides significant improvement in roll maneuverability for flight up to the maximum lift angle of attack.

* Senior Member, AIAA

Flying Qualities Engineer

This paper is declared a work of the U.S. Government and is not subject to copyright protection in the United States.

INTRODUCTION

The USAF Research Laboratory's Control Dynamics Branch has been involved in high angle-of-attack (AOA) flight research for the last several years in the areas of aircraft dynamics, aerodynamics and flight control. [6] describes a piloted simulation study of a configuration that represents the culmination of two earlier efforts[2,3] involving modeling and non-realtime simulation of aerodynamic and related flight control law modifications to a model of the baseline VISTA/F-16. Only part of the large volume of piloted simulation output data had been analyzed for inclusion of results in [6]. A complete analysis of the data has now been done and this paper presents the results, specifically focusing on the effects of the modifications on the VISTA's departure resistance and maneuverability at high-AOA (i.e. above the 29 degree AOA-limiter and up to the maximum lift AOA at 35 degrees). This introductory section briefly reviews the critical technical background concerning the modified aerodynamics, associated new control laws and the design and conduct of the piloted experiment from which the results described below were obtained.

Aerodynamic Modifications [2] describes a wind-tunnel test program conducted in 1991-2 to define the aerodynamics of an F-16 with the following aerodynamic modifications: (i) cut-back wing leading-edge extension (LEX), (ii) forebody chines and (iii) pneumatic forebody vortex control as shown in Figure 1. These will be referred to as the LEX/Chines/Blowing(LCB) modifications, with the blowing being an active control effector.

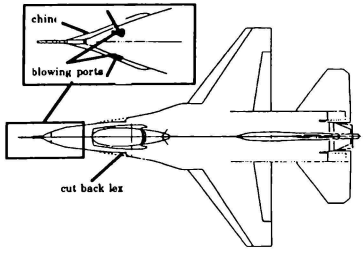


Figure 1 Active and Passive Modifications

The high-AOA effects sought were: (i) improved directional stability (due to the chines), (ii) improved longitudinal stability and elimination of the high-AOA pitch trim point (due to a cut-back LEX), and (iii) increased directional control power (due to forebody vortex control). The specific beneficial changes to the total pitching, rolling and yawing moments due to the passive (i.e., no active vortex control) modifications as well as the rolling and yawing moments resulting from the active vortex control are shown in figures 2 through 6 in [6]. In turn, these changes affect the F-16's flight characteristics as follows: (i) the F-16 tendency towards hung stall in the vicinity of sixty degrees AOA is eliminated by the effective nosedown pitching moment increment produced by reduction of the LEX area forward of the center of gravity, (ii) the increase in yawing moment above twenty-degrees AOA generated by the chines reduces the directional instability, (iii) Cut-back LEX and chines together result in an improved rolling moment for AOA greater than thirty degrees, (iv) the vortex control generates additional yawing moment, (v) the vortex control generates an associated incremental rolling moment, which is predictable and proverse through approximately thirty-five degrees AOA - however, it is adverse above thirty-five degrees and this is noted in [3] as the limiting factor in the control of sideslip.

Control Law Implementation [3] describes the development of control laws for a forebody vortex control system that augments yaw control power as the rudder loses effectiveness at high AOA. Figure 2 shows the control law block diagram from that report. The approach used in [3] was to modify the existing F-16 control laws by adding an outer feedback loop for the forebody vortex control. The outer loop control law was bang-bang with three possible states: full blowing from the left forebody slot, full blowing from the right forebody slot and no blowing. The bang-bang control law was chosen because it was considered to be the most conservative. The authors of [3] had no data that suggested linear actuation was possible and the bang-bang control law evolved naturally from the bang-bang actuator. Design of the system was accomplished using variable structure control and describing functions and is described in detail in [3]. Reference 3 also gives a detailed summary of the non-realtime simulation study of potential high-AOA flight stability and performance enhancements.

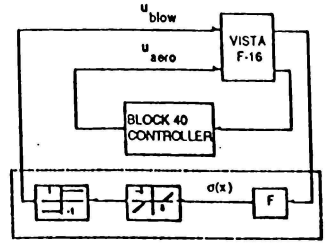


Figure 2 Forebody Blowing Control System

Just as the wind tunnel tests of [2] validated the predicted improvements in directional stability and directional control power, the results in [3] showed that use of the modifications allow the extension of the VISTA/F-16's AOA envelope to 37 degrees AOA. The same non-realtime simulation used in [3] was used in the piloted simulation study conducted in mid-September 1995 and described in detail in [6].

Experiment Set-up The piloted study used four distinct configurations:

Baseline(BASE) = baseline VISTA (29 degree AOA limiter)

Extended(EXT) = baseline configuration with limiter extended to 40 degrees

LEX/Chines(LC) = Extended configuration with cutback LEX and chines

LEX/Chines/Blowing(LCB) = LC configuration with active vortex blowing controller

EXT, LC, and LCB are the same configurations used in [3].

Pilots The pilots used for the experiment were highly qualified veteran F-16 flight test pilots assigned to the Air Force Flight Test Center at Edwards AFB and Wright-Patterson's F-22 SPO. This was considered important both for their ability to perform the experiment as designed and for their ability to comment on the results with respect to application of the enhanced capability to operational use.

Tasks The tasks required of the pilots during the study were divided into performance, flying qualities and target acquisition evaluation tasks. Tasks 1 through 3 are the same as the unmanned simulation tasks used in [3] and were repeated to provide a direct comparison to those earlier results. Task 1 defines the maximum roll rate at a given AOA for the test speed. Task 2 adds a roll attitude capture to task 1. Task 3 extends the speed range for the roll and capture dynamics as well as adding coupling dynamics from the initial angular motions. Roll acceleration and rate are the primary quantities of interest in these tests, along with the pilot assessment of the flying characteristics. Tasks 4 and 5, which are taken from the Standard Evaluation Maneuvers of [5], introduce varying degrees of difficulty to the task. This is done to examine the effect of aircraft capability on time to acquire a benignly maneuvering target. Initial conditions were chosen to assure the ability to perform the maneuvers with the AOAs of interest and in such a way as to make the task operationally relevant. The time to acquire and the maximum roll rate and acceleration are the main quantitative interest, with pilot ratings and commentary again gathered as the qualitative data. The tasks are described below.

Performance:

Task1 (Max Lateral Stick Response): From a trimmed condition, the pilot aggressively pitched the aircraft up to the desired AOA and applied a two-second full right lateral stick pulse while holding the test AOA.

Flying qualities evaluation:

Task2 (Split-S): From a trimmed condition, the pilot pitched the aircraft up to the desired test AOA (AOAt). He then rolled 180 degrees while maintaining AOA. He continued to pull through to achieve a 180 degree heading change.

Task3 (Loaded Roll Reversal): The pilot entered a 2-g right turn. He then pitched up to the desired test AOA. He then applied a left lateral stick pulse to capture approximately -60 degrees bank angle while holding the test AOA.

Target acquisition:

Target set-up: One pilot flew the baseline configuration from 1-g level trim into a constant AOA 3-g descending turn. The results were recorded and used as the target aircraft for tasks 4 and 5.

Task4 (Gross Lateral Acquisition): The test aircraft began in 1-g level flight approximately 1500 feet behind and 1000 feet below the target aircraft. When the target rolled, the pilot hesitated until the target was approximately 10 to 20 degrees off of the nose. He then quickly pulled to the test AOA, hesitated momentarily, then rolled aggressively while holding AOA to capture the target.

Task5 (Gross Longitudinal Acquisition): The test aircraft began in 1-g level flight approximately 3000 feet directly behind the target aircraft. The pilot allowed the target to reach a predetermined angle off the nose, then rolled to get into the target's maneuver plane. He then hesitated until he was in a lag position such that the test AOA occurred at target capture.

Flight Conditions The tasks mentioned above were performed for a range of trim conditions from [3] with Mach numbers between 0.3 and 0.7 and altitudes between 10,000 ft and 25,000 ft.The test AOAs ranged from 20 to 35 degrees.The combination of flight condition (Mach and altitude at trim condition) together with the desired AOAt to which the pilot was to pull define what is referred to as a case in the discussion below.

Case Definitions The 189 simulation runs done by the pilots (63 runs x 3 pilots) were organized into cases, defined in Table 1 :

Case	Task	Mach	Alt	AOAt	Configurations
1	1	.3	10k	30	EXT,LC,LCB
2	1	.3	10k	35	EXT,LC,LCB
3	2	.3	10k	30	EXT,LC,LCB
4	2	.3	10k	35	EXT,LC,LCB
5	2	.5	10k	30	EXT,LC,LCB
6	2	.5	10k	35	EXT,LC,LCB
7	2	.7	10k	30	EXT,LC,LCB
8	3	.6	25k	20	BASE,EXT,LC,LCB
9	3	.5	25k	25	BASE,EXT,LC,LCB
10	3	.5	25k	30	EXT,LC,LCB
11	3	.4	25k	35	EXT,LC,LCB
12	4	.4	24k	20	BASE,EXT,LC,LCB
13	4	.5	24k	25	BASE,EXT,LC,LCB
14	4	.5	24k	30	EXT,LC,LCB
15	4	.5	24k	35	EXT,LC,LCB
16	5	.6	25k	20	BASE,EXT,LC,LCB
17	5	.5	25k	25	BASE,EXT,LC,LCB
18	5	.5	25k	30	EXT,LC,LCB
19	5	.5	25k	35	EXT,LC,LCB

Table 1 Definition of Experiment Cases

In the conduct of the experiment the order of the configuration was randomized. Pilots were only told before starting the task whether they would be flying a configuration with a raised limiter - but not which of the three raised limiter configurations it was.

SIMULATION STUDY RESULTS

In this section results of the piloted simulation study are presented which show the beneficial effects that the modifications have on both departure resistance and maneuverability at high angles of attack. The results presented here were obtained primarily by automated simulation output data processing and analysis.Pilot commentary was recorded on audio tapes for the entire experiment. Commentary on these tapes is very consistent with the analytical results presented here.

Departure Resistance The primary objective of both the non-realtime simulation study done in [3] and this study was to determine the feasibility of expanding the VISTA/F-16's AOA envelope and thus increasing the range of AOA that it might simulate. The key to succesfully accomplishing this expansion is to enhance the vehicle's departure resistance up through the unstable 30 - 35 degree AOA region. Table 2 summarizes the occurences of departures. A departure is defined as a simulation run which contains an AOA greater than 75 degrees or a sideslip greater than 20 degrees. All departures encountered were for runs involving the EXT configuration, and all of these contained AOAs that were above 75 degrees. The highest sideslip value obtained in the study was approximately 13 degrees. As the table indicates, Task 2 accounted for all but 1 of the 10 departures(out of 189 simulation runs), with a very uniform split across pilots and Task 2 cases noted. Similarly, Table 3 summarizes the percentage of departure occurences for the entire simulation study, based on aircraft configuration. As is clearly evident, the modifications do allow the VISTA to fly to maximum lift AOA with definite increased departure resistance.

Pilot1	Pilot2	Pilot3
Task2/Case3	Task2/Case4	Task2/Case3
Task2/Case5	Task2/Case6	Task2/Case4
Task2/Case6	Task2/Case7	Task2/Case5
Task3/Case11		

Table 2 Departure Occurence cases

Config	Number of Runs	Departures	Percent
BASE	18	0	0
EXT	57	10	18
LC	57	0	0
LCB	57	0	0

Table 3 Percentage of Departures by Configuration

Maneuverability In addition to departure resistance, this study addressed the issue of whether the modifications would yield greater maneuverability of the aircraft. Note that no single uniformly accepted definition of maneuverability is in use across the aerospace community. Furthermore, it is important to realize that any such definition of maneuverability would differ still from the concept and definition of tactical utility. One approach, which is task dependent, taken towards quantifying maneuverability changes observed due to the modifications was to estimate the time it took the pilots to capture either a real target or some "target" state. This was the approach taken in [6]. Note that the results concerning maneuverability based on capture time and the PM metric described in [6] only represent a small portion of the 189 simulation runs performed during the experiment. The authors found that the graphical analysis needed for this approach (see Figures 8 and 9 of [6]) could not be readily automated, and was thus very tedious and time consuming.

The results shown there (Figures 10-16) are fairly inconclusive.

For this paper, a second approach to the quantification of maneuverability was used. This approach was used for the entire experimental matrix (all 189 simulation runs). Although perhaps not as close to the concept of tactical utility as the capture time approach, it has the advantage of being readily amenable to automation, while still having an obvious correlation with possible improvements in the vehicle's tactical utility. The second approach to defining maneuverability was to relate it to the achievement of optimal values of certain critical aircraft state parameters. These parameters were:

1) Stability axis roll rate, given by $P_s = P_b \cos\alpha + R_b \sin\alpha$ where P_s = stability axis roll rate, P_b = body axis roll rate, R_b = body axis yaw rate, α = angle-of-attack

2) Stability axis roll acceleration

3) Loaded body axis roll rate, given by $P_{bL} = (P_b\ n_z\ g)/\cos\alpha$ where P_{bL} = loaded body axis roll rate, n_z = load factor, g = gravitational acceleration

4) Sideslip

5) Pitch rate

6) Pitch acceleration

These are referred to as maneuverability parameters. Although sideslip is not typically classified as a maneuverability parameter, it is included in the group because (as the results will presently show) it is highly correlated with lateral maneuverability, as measured by the first three corresponding maneuverability metrics defined below:

1) Maximum stability axis roll rate attained during a simulation run

2) Maximum stability axis roll acceleration attained during a simulation run

3) Maximum loaded body axis roll rate attained during a simulation run

4) Maximum sideslip angle attained during a simulation run

5) Maximum pitch rate attained during a simulation run

6) Maximum pitch acceleration attained during a simulation run

These 6 maneuverability metrics were calculated for the entire experimental design matrix. Figures 3 through 10 contain plots summarizing the results.

The analysis done and plots generated form a systematic attempt to show the effects of these modifications on the 6 maneuverability metrics defined above. Although both individual pilot results and pilot-averaged results and plots were obtained, for conciseness only the pilot-averaged plots are displayed here. Since individual pilot differences were noted during the execution of similarly defined tasks, the pilot-averaged results are assumed to be the best estimate of the effects that the modifications had on the maneuverability metrics. Likewise, results were tabulated for maneuverability both accounting and not accounting for departures. If any pilot departed on a case's run, the pilot averaged result was considered to be a departure. For the metrics chosen, the expected trends were that metrics 1-3 would increase for a configuration change from EXT to LC and from LC to LCB. Metric 4(maximium sideslip attained during the run) would decrease, due to the chines and blowing, with this causing the improved roll predicted to be seen in metrics 1-3. Since the cutback LEX was intended to decrease pitch up tendency, a slight decrease in pitch rate and acceleration was expected in LC and LCB compared to BASE and EXT.

Maneuverability Metric Results (Pilot-avg Summary Plots): The plots shown in figures 3 through 8 summarize pilot-averaged results for all configurations, all 19 cases and each of the 6 maneuverability metrics.Circled markers indicate departures. Based on the metric definitions, it is intuitively clear that for metrics other than metric 4, the more often the LC and LCB values are highest on the plot, the more that indicates enhanced maneuverability attributed to the modifications. A close scanning of these plots indicates that the modifications definitely tend to yield more case results with higher roll rates and

accelerations and lower max sideslip values. But the EXT configuration has a predominance of highest pitch rate and acceleration values. However, as expected , there were several departures for the EXT configuration. Likewise, as noted above, the cutback LEX would be expected to produce less pitch up.

Maneuverability Metric Results(Configuration Comparisons Not Accounting For Departures): Here a simple tabulation was done to see how many times each configuration got the best value for the various 19 cases. Here "best" means highest , except for metric 4 (maximium sideslip attained during the simulation run). Departure was not considered in the tabulation. For the pilot-averaged plots shown in Figures 3 through 8 and the 6 maneuverability metrics, respectively, Figure 9 indicates the percentage of cases for which each configuration had the best value. Thus, for example, the LCB configuration obtained the best value of metric 1(maximum stability axis roll rate attained during the run) for more than 50 percent of the 19 cases. Likewise, the LCB configuration obtained the best (lowest) value of metric 4 for more than 60 percent of the 19 cases. The EXT configuration obtained the best value for metrics 3,5 and 6 for the highest percentage of the cases. However, the results shown in Figure 9 were obtained not Accounting For Departures(AFD).

Maneuverability Metric Results(Configuration Comparisons AFD): In Figure 10 departures are accounted for by simply defining the second-best value obtained as the best if the best value was for a departed run. Here the modified configurations LC and LCB obtain considerably higher percentages of best values for the pitch maneuverability metrics relative to the EXT. Indeed, as Figure 10 shows, if only non-departed simulation run values are allowed to be considered best , then the LC and especially the LCB configuration shows striking predominance in the percentage of cases for which it received the best value for ALL maneuverability metrics(only tieing with the BASE configuration for most bests in pitch acceleration).

Figure 10 shows that roll maneuverability is enhanced, sideslip attenuation is enhanced and pitch maneuverability is certainly not degraded by making the LC and LCB modifications. Table 3 below gives the amount of enhancement in the VISTA's high-AOA maneuverability produced by the modifications.

Amount of maneuverability enhancement produced by modifications: To arrive at these estimates, only the pilot-averaged results for non-departed runs were used, since a "bottom-line" estimate of enhancement should not consider departures as contributing to enhancement. Since Figure 10 shows that for non-departed runs, the LCB configuration was best for all metrics the majority of the time (only tieing with the BASE configuration for most bests in pitch acceleration), the numbers in Table 4.3 are obtained by averaging the percentage of enhancement (increase in the metric for all except metric 4) over all the cases for which the LCB was best. This was done for LCB relative to each of the other 3 configurations.

Metric 1	LCB over LC	17%
	LCB over EXT	46%
	LCB over BASE	8%
Metric 2	LCB over LC	4%
	LCB over EXT	11%
	LCB over BASE	14%
Metric 3	LCB over LC	20%
	LCB over EXT	22%
	LCB over BASE	4%
Metric 4	LCB over LC	32%
	LCB over EXT	27%
	LCB over BASE	30%
Metric 5	LCB over LC	14%
	LCB over EXT	13%
	LCB over BASE	5%
Metric 6	LCB over LC	9%
	LCB over EXT	20%

Table 4.3 Average percentage of maneuverability enhancement due to LCB

Note that no comparison between LCB and BASE is possible for metric 6, since for the cases where LCB was highest in pitch acceleration the BASE configuration was not flown. Indeed, it may appear that, with the exception of metric 4 (maximum sideslip), the BASE configuration performed quite well versus the LCB based on this table. But recall that the AOAs tested for the BASE were 20 and 25 degrees, and only 6 cases of the 19 were so tested. Also, one might claim that the EXT performed comparably to the LC based on these numbers. But ALL departures seen were for the EXT configuration and it departed nearly 20% of the time.

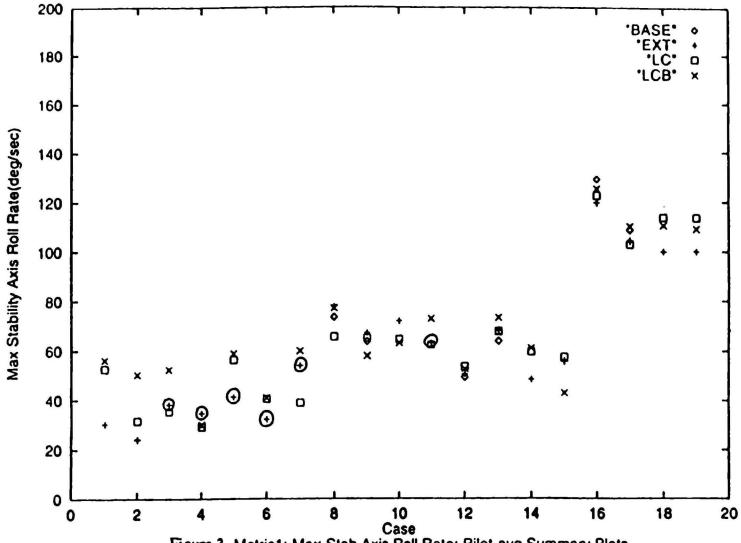


Figure 3 Metric1: Max Stab Axis Roll Rate: Pilot-avg Summary Plots

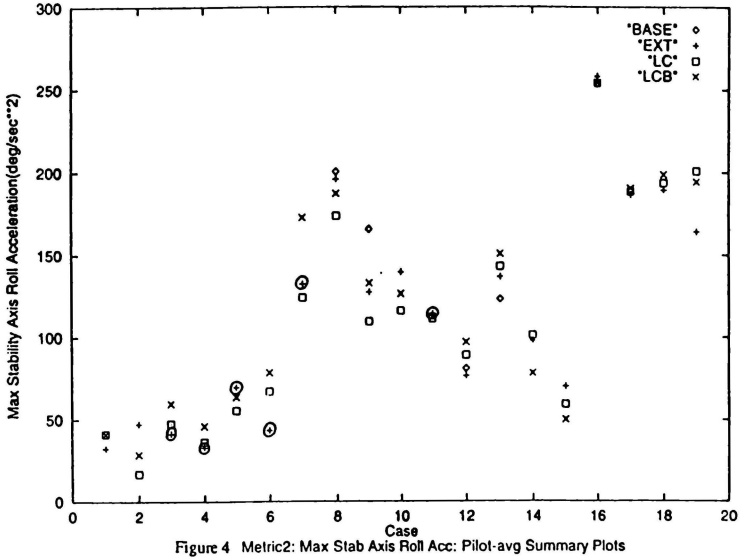


Figure 4 Metric2: Max Stab Axis Roll Acc: Pilot-avg Summary Plots

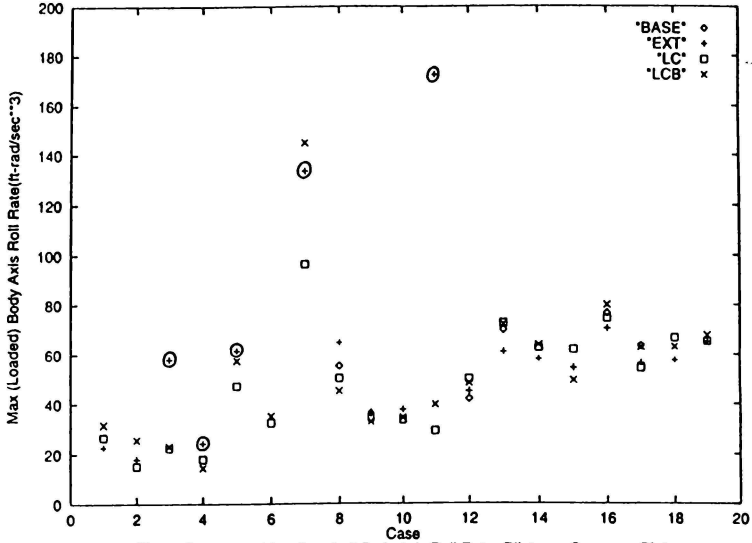


Figure 5 Metric3: Max (Loaded) Body Axis Roll Rate: Pilot-avg Summary Plots

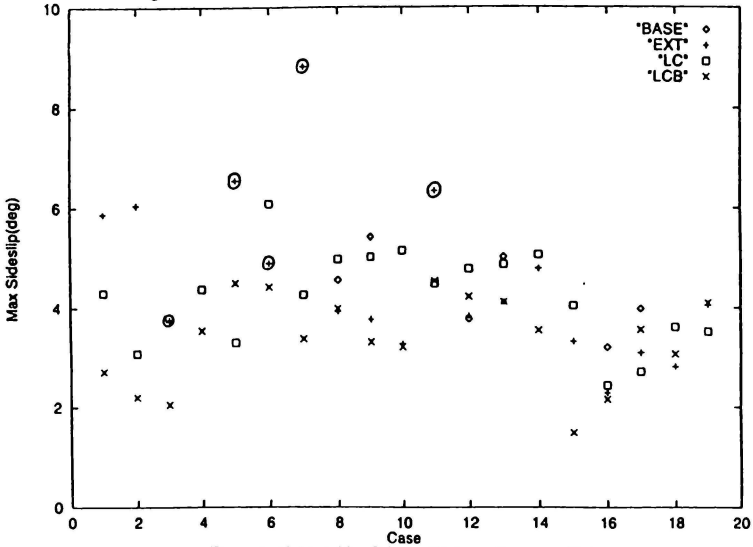


Figure 6 Metric4: Max Sideslip: Pilot-avg Summary Plots

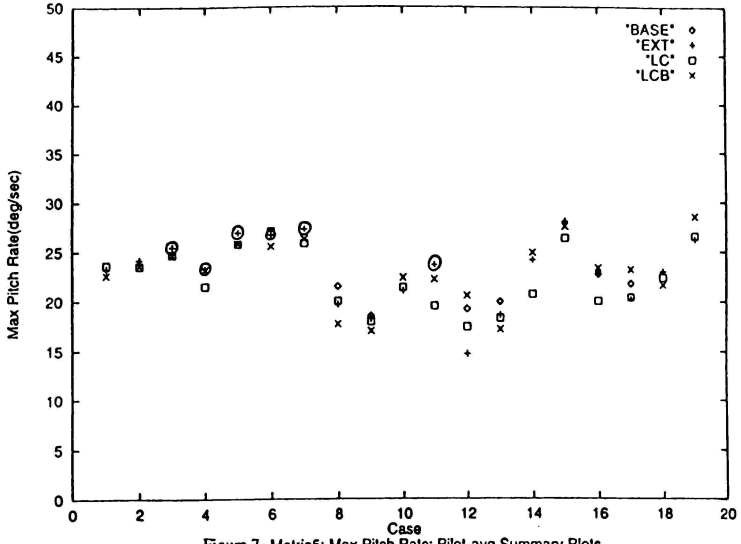


Figure 7 Metric5: Max Pitch Rate: Pilot-avg Summary Plots

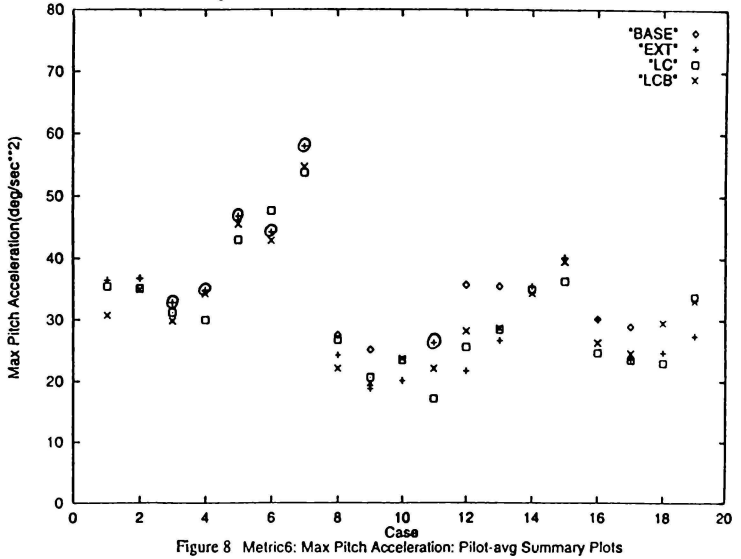


Figure 8 Metric6: Max Pitch Acceleration: Pilot-avg Summary Plots

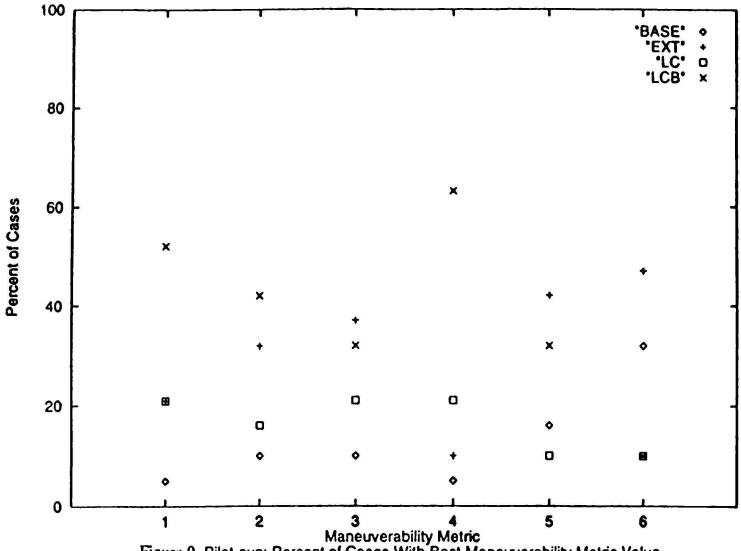


Figure 9 Pilot-avg: Percent of Cases With Best Maneuverability Metric Value

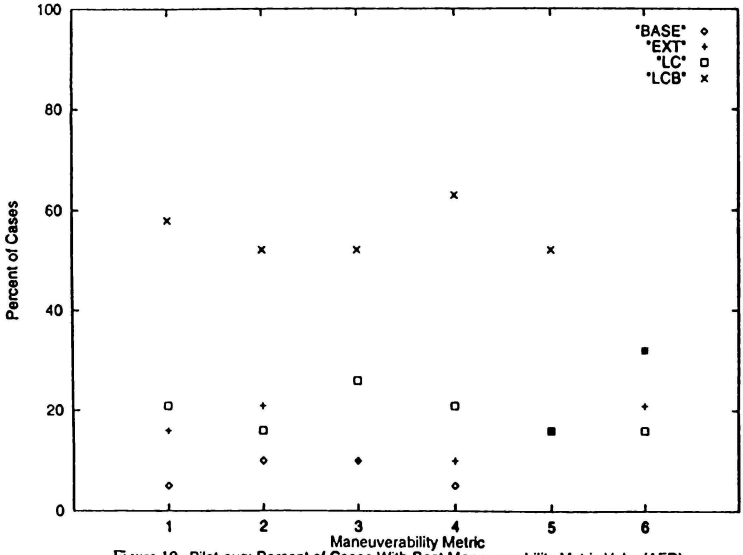


Figure 10 Pilot-avg: Percent of Cases With Best Maneuverability Metric Value(AFD)

CONCLUSIONS

Based on a thorough analysis of the simulation output data, the conclusion is that the use of all of these modifications allows for flight up to the maximum lift AOA(approximately 35 degrees) with greatly increased departure resistance compared to a VISTA without the modifications. Further, the modifications provide a significant (27%) improvement in sideslip attenuation over an unmodified VISTA flying at max lift AOA, and a significant (46%) improvement in roll maneuverability as measured by the maximum stability axis roll rate. The modifications produce pitch maneuverability at least comparable to an unmodified VISTA.

The issue of tactical utility improvements due to the modifications was outside the scope of the experiment. However, the results obtained concerning departure resistance and improved roll maneuverability at max lift AOAs suggest that such tactical utility improvements would be consistent with the results of this study. Further research and experimentation, including one-on-one piloted simulation combat scenarios between these 4 configurations, would be the next step in trying to determine possible tactical utility improvements due to these modifications. Following that, similar combat scenario simulations between this modified VISTA and the MATV could also prove to be very beneficial in determining the usefulness of forebody vortex control in air-to-air combat.

REFERENCES

[1] McKeehen, P. D., "F-16/VISTA High-Angle-of-Attack Aerodynamic Modeling and Analysis", WL-TM-91-304, Wright Laboratory, Wright-Patterson AFB OH, February 1991.

[2] Simon, J. M., LeMay, S., Brandon, J. M., "Results of Exploratory Wind Tunnel Tests of F-16/VISTA Forebody Vortex Control Devices", WL-TR-93-3013, Wright Laboratory, Wright-Patterson AFB OH, January 1993.

[3] Adams, R. J., Buffington, J. M., "Design and Analysis of Modifications for VISTA F-16 High Angle-of-Attack Envelope Expansion", WL-TR-93-3064, Wright Laboratory, Wright-Patterson AFB OH, July 1993.

[4] McKeehen, P.D. ,"GENESIS Simulation of a Modified VISTA/F-16",AIAA-95-3381-CP, presented at AIAA Flight Simulation Technologies Conference, August 1995.

[5] Wilson, D.J., Riley, D.R., and Citurs, K.D., "Aircraft Maneuvers for the Evaluation of Flying Qualities and Agility - Maneuver Descriptions and Selection Guide," WL-TR-93-3082, August 1993.

[6] McKeehen, P.D., Cord, T.J., Nguyen, B.T.,"Modified VISTA/F-16 Piloted Simulation Study", AIAA-96-3508-CP, presented at AIAA Flight Simulation Technologies Conference, July 1996.

FUZZY LOGIC BASED FUNCTION MODELING IN AERODYNAMIC SIMULATIONS

Douglas D. Hensley
Raytheon Aircraft Company, MS: 931/B-31
9709 E. Central
Wichita, Kansas 67206
E-Mail: douglas_hensley@raytheon.com

Abstract

This paper presents the background for an investigation into the use of fuzzy logic based function modeling as an alternative to linear table look-up and interpolation in flight simulation. Fuzzy systems have been shown to belong to the class of functions which are capable of approximating any real function to any desired degree of accuracy. These fuzzy systems are built upon very simple and efficient mathematical functions and can be implemented in a variety of forms.

For accurate fuction modeling in a time critical environment, the Sum-Product, Sum-Mean and Time-Critical Optimized inferencing methods would be appropriate choices. Implementing these inferencing methods as fuzzy clustering rules with the output consequent defined as a linear function of the input variables will significantly improve the accuracy, flexibility and size of the resulting function model.

Introduction

Typically function modeling in real-time flight simulation is performed as a linear table look-up and interpolation in N dimensions, where N represents the number of independent variables used to define the function model. The linear interpolation method uses mathematically simple operators and can be efficiently implemented. Modeling resolution in non-linear regions of the function being modeled is achieved through denser breakpoint spacing along the independent variable's domain. One drawback of the linear table look-up is the 2^N exponential growth in the number of calculations required to perform a multi-parameter interpolation.

Fuzzy logic presents a method of function modeling that deserves consideration as a possible alternate to the linear table look-up and interpolation method. This paper presents the background for an investigation into the use of fuzzy logic based function modeling in aerodynamic simulations.

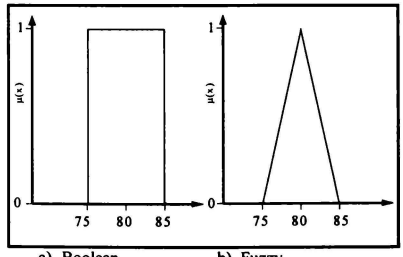
Fuzzy Logic Primer

Fuzzy Sets

Fuzzy logic is built upon fuzzy set theory, which is itself a generalization of boolean set theory.[1,2] The characteristic function, or membership function, of set theory indicates the degree of membership of a given value in that set. For a boolean set, A, the characteristic function would be defined as:

$$CA(x) = \begin{cases} 1, \text{ when } x \in A \\ 0, \text{ otherwise} \end{cases} \quad (1)$$

In fuzzy set theory, the characteristic function is generalized to a membership function that assigns a degree of membership ranging from 0 (complete exclusion) to 1 (complete membership). The degree of membership of an input value, x, would be denoted $\mu(x)$. Example characteristic functions for a boolean and a fuzzy set representing the linguistic term, "Comfortable Room Temperature," where 80 degrees is considered the most desirable room temperature are shown in Figure 1.



a) Boolean b) Fuzzy

Figure 1: Characteristic Function for Comfortable Room Temperature

The membership of the boolean set remains maximum for all room temperatures within the set's domain, and

Copyright © 1997 by The Raytheon Aircraft Company. Published by the American Institute of Aeronautics and Astronautics, Inc. with permission.

abruptly becomes minimum at the set boundaries. Thus 75.01 degrees is considered equally comfortable as 80 degrees, while 74.99 degrees is no longer regarded as comfortable. In contrast, the fuzzy set more clearly represents the concept denoted by "Comfortable Room Temperature." The membership in the fuzzy set is maximum at the desired temperature and gradually decreases at temperatures further from the desired temperature.

Classical Fuzzy Set Operations

The boolean set theory definitions for the set operations, intersection (denoted $\cap$, $\wedge$, AND), union (denoted $\cup$, $\vee$, OR) and complement (denoted NOT) are depicted in Table 1:

Membership Values		Inter-section	Union	Complement	
A	B	$A \cap B$	$A \cup B$	$\overline{A}$	$\overline{B}$
0	0	0	0	1	0
0	1	0	1	1	0
1	0	0	1	0	1
1	1	1	1	0	1

Table 1: Boolean Set Operations

These set operations are most commonly defined in fuzzy set theory as[1,2]:

Intersection: $\mu_A(x) \cap \mu_B(x) = \text{Min}(\mu_A(x), \mu_B(x))$
Union: $\mu_A(x) \cup \mu_B(x) = \text{Max}(\mu_A(x), \mu_B(x))$
Complement: $\mu_{\overline{A}}(x) = 1 - \mu_A(x)$

This definition of fuzzy set operators is commonly referred to as the classical fuzzy operators. Examples of these classical set operations for representative fuzzy sets are shown in Figure 2. It should be noted that at the set membership boundaries, when $\mu(x) = 0$ or $\mu(x) = 1$, the fuzzy set operations yield the same values as the boolean set operations shown in Table 1.

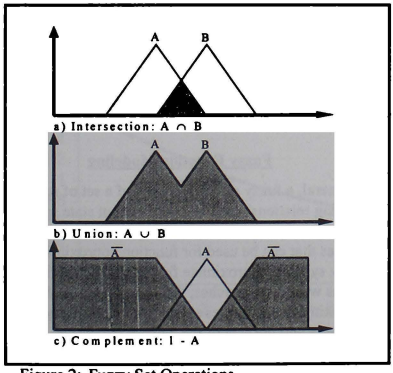


Figure 2: Fuzzy Set Operations

Fuzzy Inferencing

Relationships between fuzzy sets of several variables can be expressed in terms of IF-THEN rules defining the fuzzy implications of a fuzzy system. These rules would take the general form[1,2]:

$$\text{If } X_1 \text{ is } A_1 \ \&\ X_2 \text{ is } A_2 \ \&\ \ldots \text{ Then } Z_1 \text{ is } B_1 \quad (2)$$

Where: A_1, A_2, and B_1 are fuzzy sets.

The rule consequent, B_1 of this general rule form could be assigned a singleton, or constant value. This rule consequent could alternatively be defined as a linear function of the input variables[3]:

$$\text{If } X_1 \text{ is } A_1 \ \&\ X_2 \text{ is } A_2 \ \&\ \ldots \text{ Then } Z_1 = C_0 + C_1X_1 + C_2X_2 + \ldots \quad (3)$$

An example fuzzy system for a simple temperature controller is presented in Figure 3.

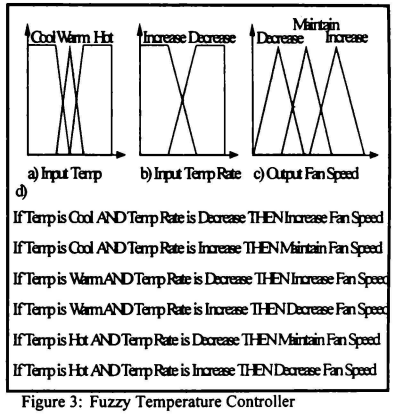


Figure 3: Fuzzy Temperature Controller

Using the classical fuzzy set operations with AND (set intersection) defined as the Min operation and OR (set union) defined as the Max operation, the Max-Min inferencing method would evaluate these fuzzy implications as:

$$\text{Max}[\ \text{Min}(\mu_A(x_1), \mu_B(x_1)), \text{Min}(\mu_A(x_2), \mu_B(x_2))] \quad (4)$$

The inferencing results using representative input values for the temperature and the rate of change of the temperature are depicted in Figure 4.

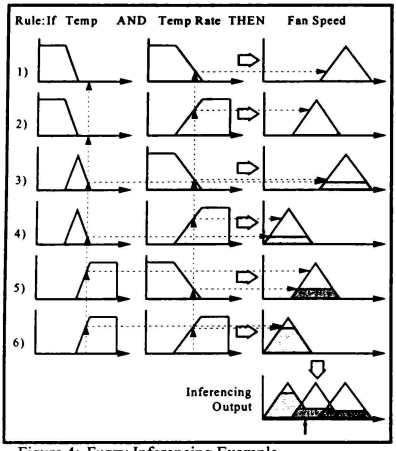


Figure 4: Fuzzy Inferencing Example

Non-Classical Fuzzy Inferencing

With the fuzzy set theory relaxation of membership grades to include values between 0 and 1, a variety of fuzzy set operators satisfying the boolean set boundary conditions and other fuzzy theory axiomatic conditions have been defined.[1,2] The choice of fuzzy set operators and resulting inferencing method can significantly affect the characteristics of a fuzzy system. As an example, the Max-Min inferencing method of Equation 4 is relatively insensitive to variations in the input values. For a given value of X_1, any value of X_2 such that $\mu(X_1) \leq \mu(X_2)$, has no effect on the resulting inferencing output. This characteristic is unacceptable for most function modeling applications.

A large number of non-classical operators exist that overcome the input value variation insensitivity of the classical Max-Min inferencing method. The available number of non-classical operators appropriate for function modeling applications is considerably reduced when applicability within a time-critical environment is a factor. Some operators and inferencing methods that satisfy these conditions would include the Sum-Product[1,2], Sum-Mean[1,2], and Time-Critical Optimized[4] inferencing methods. These inferencing methods are defined respectively as:

$$(\mu_A(x_1) \times \mu_B(x_1)) + (\mu_A(x_2) \times \mu_B(x_2)) \quad (5)$$

$$(\mu_A(x_1) + \mu_B(x_1)) / 2 + (\mu_A(x_2) + \mu_B(x_2)) / 2 \quad (6)$$

$$(\text{Min}(\mu_A(x_1), \mu_B(x_1)) \times \text{Mean}(\mu_A(x_1), \mu_B(x_1)) + \text{Min}(\mu_A(x_2), \mu_B(x_2)) \times \text{Mean}(\mu_A(x_2), \mu_B(x_2))) / 2 \quad (7)$$

Output Defuzzification

The output for the fuzzy system shown in Figure 4 is in the form of a fuzzy set. This output set indicates a need to decrease the fan speed. Clearly a fan controller must output a specific value for changing the fan speed. Thus a final step, defuzzification, must be performed. The most common form of defuzzification is to find the center-of-area or center-of-gravity of the output fuzzy set.[5,6] Defuzzification of the Figure 4 output set by the center-of-area method yields the single output value indicated by the arrow.

For the purpose of function modeling, the Quality method of defuzzification possesses very desirable properties.[6] This defuzzification method is defined as:

$$\text{Output} = \left[\sum_i (C_i \times W_i)\right] \Big/ \sum_i (W_i) \quad (8)$$

Where: $W_i = h_i / d_i$

The terms C_i, h_i, and d_i for Equation 8 are defined in Figure 5.

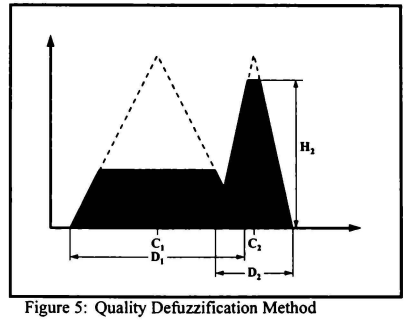


Figure 5: Quality Defuzzification Method

Fuzzy Function Modeling

In general, a fuzzy system consists of a set of rules defining relationships between the input state space and the output state space. These fuzzy rules form fuzzy patches that can be used for function approximation. Fuzzy systems approximate functions by covering their graphs with fuzzy patches; the accuracy of the function approximation increases as the number of fuzzy patches increases and the size of the patches decreases. Fuzzy systems have been demonstrated to belong to the class of functions which are capable of approximating any real function to any desired degree of accuracy.[7]

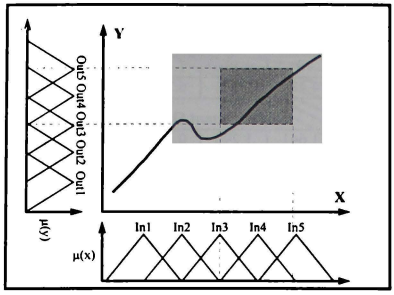


Figure 6: Fuzzy State Space Rule Patch

Function Modeling: Fuzzy Associative Memory

The fuzzy membership sets for each input variable, X, and output variable, Y, quantize the input-output state space into a cartesian product, X×Y, of potential fuzzy rule combinations. These potential fuzzy rules form fuzzy patches that can be used for function approximation. The potential fuzzy rule combinations formed by the quantization of the input-output state space is called the Fuzzy Associative Memory (FAM).[8] The FAM method is the most common method used to implement fuzzy systems.

Figure 6 depicts the manner in which a FAM rule defines a fuzzy patch on the input-output state space of a function. The fuzzy sets for the input variable, X, are shown across the domain of X. The fuzzy sets for the output variable, Y, are shown across the range of Y. These membership sets quantize the input-output state space X×Y, into potential fuzzy rule combinations. The shaded region of Figure 6 represents the fuzzy patch for the rule, "If X is In4 Then Y is Out4."

Function Modeling: Fuzzy Clustering

In the fuzzy clustering approach to function modeling, the potential fuzzy rule combinations are not based upon the cartesian product quantization of the input-output state space through universally defined input and output fuzzy membership sets. Instead, each rule includes the definition of fuzzy input and output membership sets which are unique to that specific rule. Thus the state space is partitioned into arbitrarily positioned fuzzy patches. Each fuzzy rule still defines relationships between the input state space and the output state space, but only for a region of the state space uniquely defined by that rule's membership sets. This approach to function modeling yields great flexibility and can improve the accuracy and size of the resulting function model.[9]

Function Modeling: Examples

Representative aerodynamic functions with one and two independent parameters respectively, were selected to demonstrate examples of fuzzy logic based function modeling. In the discussion of these modeling examples, it should be noted these results are not optimized. The measures of modeling error presented should be considered conceptual measures rather than literal measures of merit for each method of fuzzy modeling demonstrated.

Function Modeling: 1-D Function The Subsonic Wing Lift-Curve Slope[10] of Figure 7 was selected as a representative aerodynamic function with one independent parameter (1-D function). For a 1-D function, the Sum-Product and Sum-Mean inferencing methods of Equations 5 and 6, reduce to equivalent

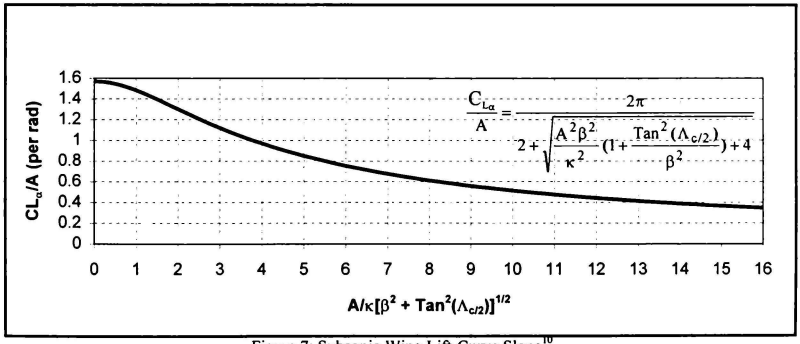


Figure 7: Subsonic Wing Lift-Curve Slope[10]

implementations. Additionally, because the number of input fuzzy sets and consequents are the same, the FAM method and fuzzy clustering approach yield equivalent fuzzy rule definitions.

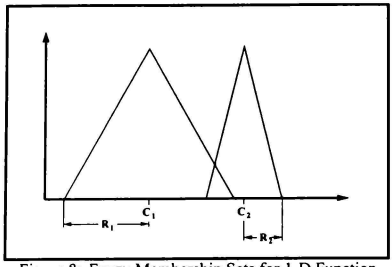


Figure 8: Fuzzy Membership Sets for 1-D Function

Figure 8 depicts the manner in which the input fuzzy sets were implemented for this example 1-D function. The fuzzy set membership values were calculated as:

$$\mu(x) = \text{Max}(1 - |C_n - x| / R_n, 0) \quad (9)$$

The results for the non-optimized fuzzy models of this function using the Equation 2 and Equation 3 rule representations are presented in Table 2. The error term for each model was calculated as the sum of the square of the difference between the Figure 7 function and the respective Table 2 model sampled at an interval of 0.2. While the range of error for the inferencing methods are similar, the rule form of Equation 3 allows the use of fewer fuzzy rules to achieve this similar level of modeling accuracy.

Function Modeling: 2-D Function The Wing Lift-Curve-Slope Thickness Correction Factor[10] of Figure 9 was selected as a representative aerodynamic function with two independent parameters (2-D function).

Figures 10 and 11 depict the input fuzzy set representation used for this example 2-D function. The Figure 10 triangular fuzzy set membership values were used for the FAM rule form and were calculated as:

$$\text{If } x < C_n \quad \mu(x) = \text{Max}[1 - (C_n - x)/(C_n - C_{n-1}), 0]$$
$$\text{Else} \quad \mu(x) = \text{Max}[1 - (x - C_n)/(C_{n+1} - C_n), 0] \quad (10)$$

Sum-Product/Sum-Mean as Equation 2					
Input Membership Sets					
C	0	4.133	7.433	8.774	13.462
R	4.485	5.413	6.733	6.871	3.665
Consequent Outputs					
C_0	1.641	1.290	0.475	0.549	0.342
Error:	0.0016				

Sum-Product/Sum-Mean as Equation 3			
Input Membership Sets			
C	0	9.085	12.657
R	15.838	7.786	12.423
Consequent Outputs			
C_0	0.074	0.074	0.074
C_1	1.344	0.344	0.041
Error:	0.0022		

Time-Critical Optimized as Equation 2					
Input Membership Sets					
C	0.016	0.703	8.115	13.253	15.834
R	9.032	10.438	15.901	6.729	5.085
Consequent Outputs					
C_0	2.000	1.854	0.803	0.462	0.375
Error:	0.0024				

Time-Critical Optimized as Equation 3			
Input Membership Sets			
C	0	13.116	16
R	16	12.172	5.760
Consequent Outputs			
C_0	0.135	0	0
C_1	1.447	0.471	0.222
Error:	0.0044		

Table 2: Fuzzy Models of a 1-D Function

The fuzzy clustering implementation used the Figure 11 gaussian fuzzy set membership values; these membership values were calculated as:

$$\mu(x) = e^{-0.5[(C_n - x)/\sigma_n]^2} \quad (11)$$

	$C_{Y1} = 0.04$	$C_{Y2} = 0.08$	$C_{Y3} = 0.12$	$C_{Y4} = 0.16$	$C_{Y5} = 0.20$	$C_{Y6} = 0.24$	$C_{Y7} = 0.28$
In1	Out5	Out5	Out5	Out5	Out5	Out5	Out5
In2	Out5	Out5	Out4	Out4	Out3	Out3	Out3
In3	Out5	Out4	Out3	Out3	Out2	Out2	Out2
In4	Out4	Out4	Out3	Out2	Out2	Out1	Out1
In5	Out4	Out4	Out3	Out3	Out2	Out2	Out1

Table 3: FAM Rule Input-Output Consequent Assignments

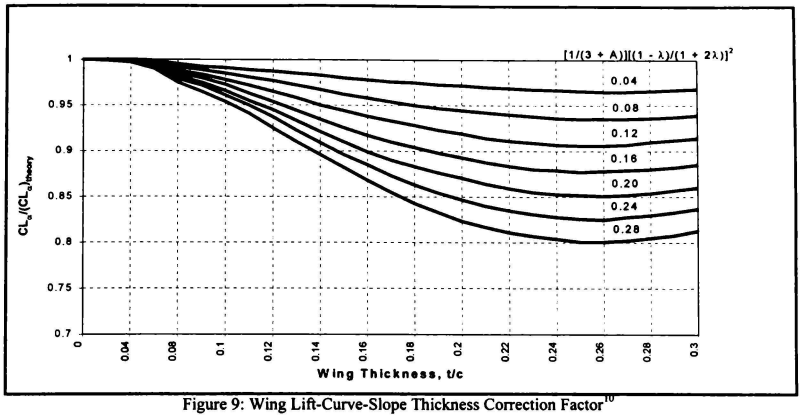


Figure 9: Wing Lift-Curve-Slope Thickness Correction Factor[10]

The FAM models of this 2-D function were defined using the input-output rule relationships shown in Table 3. The fuzzy sets for the second independent parameter (defining the separate curves) were defined using the Figure 10 and Equation 10 form with the centers at the parameter's breakpoint values (C_{Yn} = 0.04, 0.08, 0.12, 0.16, 0.20, 0.24, and 0.28).

The results for the non-optimized FAM fuzzy models and fuzzy clustering models of this function using the Equation 2 and Equation 3 rule representations are presented in Tables 4 and 5 respectively. The error term for each model was calculated as the sum of the square of the difference between the Figure 9 function and the respective Table 4 or Table 5 model sampled at an interval of 0.01.

In each table, one or more models yielded a noticeably larger level of modeling error than the other models of that table. This is due to optimizing software deficiencies rather than deficiencies in the capabilities of the fuzzy modeling. In general, the FAM models demonstrated nearly an order of magnitude greater error than the fuzzy clustering models. This significantly greater error for the FAM models can be attributed to two sources. First, all of the FAM models used a fixed rule base that was likely not optimal for any of the example models. Second, the FAM method employs global fuzzy membership sets resulting in a compromised membership set definition to reduce the total modeling error. It should also be noted that the FAM models employed 35 rules, while the fuzzy clustering models consisted of 21 rules and still achieved an order-of-magnitude improvement in modeling accuracy.

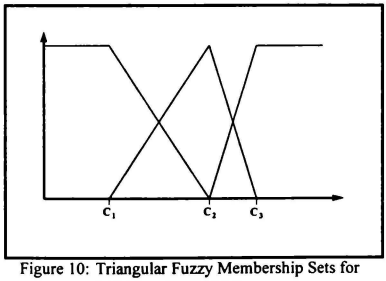


Figure 10: Triangular Fuzzy Membership Sets for 2-D Function FAM Form

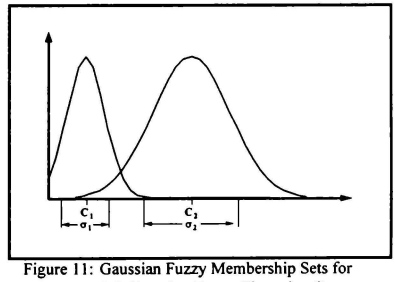


Figure 11: Gaussian Fuzzy Membership Sets for 2-D Function Fuzzy Clustering Form

Sum-Product as Equation 2											
	Input Membership Sets						Consequent Outputs				
	In1	In2	In3	In4	In5		Out1	Out2	Out3	Out4	Out5
C	0.070	0.137	0.174	0.255	0.3	C_0	0.805	0.859	0.912	0.948	0.992
Error:	0.0185										

Sum-Product as Equation 3											
	In1	In2	In3	In4	In5		Out1	Out2	Out3	Out4	Out5
C	0	0.137	0.166	0.266	0.3	C_0	0.747	0.862	0.907	0.951	0.981
						C_1	0.222	0	0	0	0
Error:	0.0267										

Sum-Mean as Equation 2											
	In1	In2	In3	In4	In5		Out1	Out2	Out3	Out4	Out5
C	0	0.066	0.167	0.260	0.3	C_0	0.800	0.870	0.930	0.970	1.000
Error:	0.2863										

Sum-Mean as Equation 3											
	In1	In2	In3	In4	In5		Out1	Out2	Out3	Out4	Out5
C	0.059	0.125	0.172	0.265	0.3	C_0	0.596	0.790	0.894	0.956	1.000
						C_1	0	0	0	0.3	0.3
Error:	0.0438										

Time-Critical Optimized as Equation 2											
	In1	In2	In3	In4	In5		Out1	Out2	Out3	Out4	Out5
C	0.067	0.142	0.168	0.256	0.3	C_0	0.807	0.861	0.912	0.948	0.992
Error:	0.0203										

Time-Critical Optimized as Equation 3											
	In1	In2	In3	In4	In5		Out1	Out2	Out3	Out4	Out5
C	0	0.157	0.162	0.258	0.3	C_0	0.769	0.888	0.938	0.976	1.000
						C_1	0.142	-0.110	-0.114	-0.128	-0.074
Error:	0.0280										

Table 4: FAM Fuzzy Models of a 2-D Function

Summary

This paper presents the background for an investigation into the use of fuzzy logic based function modeling as an alternative to linear table look-up and interpolation in a flight simulation. Fuzzy systems have been shown to belong to the class of functions which are capable of approximating any real function to any desired degree of accuracy. These fuzzy systems are built upon very simple and efficient mathematical functions and can be implemented in a variety of forms. The choice of fuzzy set operators and resulting inferencing method can significantly affect the characteristics of a fuzzy system.

For accurate fuction modeling in a time critical environment, the Sum-Product, Sum-Mean and Time-Critical Optimized inferencing methods would be appropriate choices. Implementing these inferencing methods as fuzzy clustering rules with the output consequent defined as a linear function of the input variables will significantly improve the accuracy, flexibility and size of the resulting fuzzy logic based function model.

A future paper will discuss implementation issues and timing comparisons between the fuzzy logic based function modeling systems presented in this paper and linear table look-up and interpolation. Ultra-fast, non-conventional fuzzy systems just emerging within the fuzzy logic community will also be examined as a potential function modeling method in aerodynamic simulations.

Sum-Product as Equation 2																				Error: 0.0077	
	C_{Y1} = 0.04			C_{Y2} = 0.08			C_{Y3} = 0.12			C_{Y4} = 0.16			C_{Y5} = 0.20			C_{Y6} = 0.24			C_{Y7} = 0.28		
C	0.079	0.133	0.278	0.000	0.297	0.300	0.088	0.132	0.253	0.000	0.295	0.300	0.004	0.158	0.171	0.000	0.300	0.300	0.033	0.137	0.171
R	0.030	0.030	0.030	0.030	0.030	0.030	0.030	0.030	0.030	0.030	0.030	0.030	0.030	0.030	0.030	0.030	0.030	0.030	0.030	0.030	0.030
C_0	1.000	0.975	0.967	1.000	1.000	0.868	1.000	0.933	0.910	0.989	0.925	0.829	0.999	1.000	0.806	0.981	0.839	0.829	0.996	0.957	0.800
Sum-Product as Equation 3																				Error: 0.3748	
C	0.000	0.147	0.300	0.000	0.058	0.300	0.000	0.055	0.300	0.000	0.049	0.300	0.000	0.047	0.300	0.000	0.040	0.300	0.000	0.060	0.300
R	0.153	0.202	0.077	0.110	0.192	0.044	0.108	0.201	0.053	0.105	0.196	0.063	0.108	0.210	0.056	0.113	0.214	0.059	0.008	0.002	0.300
C_0	1.032	0.968	0.973	1.074	0.937	0.965	1.124	0.891	0.951	1.167	0.853	0.924	1.215	0.805	0.910	1.283	0.741	0.898	0.700	1.039	1.085
C_1	-0.012	-0.117	0.017	-0.065	-0.114	-0.055	-0.082	-0.110	-0.075	-0.084	-0.109	-0.078	-0.084	-0.112	-0.077	-0.084	-0.115	-0.077	-0.096	-1.144	-1.159
Sum-Mean as Equation 2																				Error: 0.1258	
C	0.101	0.066	0.300	0.058	0.086	0.300	0.035	0.062	0.288	0.025	0.244	0.300	0.016	0.123	0.300	0.014	0.268	0.300	0.011	0.198	0.300
R	0.020	0.020	0.020	0.020	0.020	0.020	0.020	0.020	0.020	0.020	0.020	0.020	0.020	0.020	0.020	0.020	0.020	0.020	0.020	0.020	0.020
C_0	1.000	1.000	0.984	1.000	1.000	0.896	1.000	1.000	0.846	1.000	0.800	0.972	1.000	0.947	0.800	1.000	0.800	0.834	1.000	0.800	0.800
Sum-Mean as Equation 3																				Error: 0.0011	
C	0.000	0.064	0.300	0.001	0.080	0.239	0.000	0.077	0.300	0.000	0.071	0.281	0.000	0.101	0.153	0.000	0.059	0.300	0.000	0.059	0.151
R	0.010	0.038	0.012	0.010	0.011	0.013	0.036	0.011	0.022	0.060	0.031	0.022	0.074	0.046	0.121	0.075	0.065	0.053	0.012	0.043	0.122
C_0	1.200	0.922	0.796	1.200	0.792	0.953	0.919	1.072	0.932	0.991	0.824	1.108	1.033	1.200	0.700	1.200	0.950	0.792	1.000	1.200	0.961
C_1	0.962	0.949	0.906	1.149	1.191	-0.399	1.227	0.803	-0.688	1.497	-0.460	-0.404	1.127	-1.087	-0.093	1.194	-1.127	-0.773	-0.100	-0.498	-1.139
Time-Critical Optimized as Equation 2																				Error: 0.0048	
C	0.101	0.300	0.300	0.000	0.244	0.245	0.055	0.160	0.178	0.000	0.003	0.256	0.127	0.137	0.300	0.137	0.147	0.300	0.017	0.019	0.286
R	0.021	0.022	0.026	0.019	0.025	0.024	0.022	0.023	0.023	0.020	0.020	0.022	0.021	0.021	0.021	0.021	0.020	0.019	0.016	0.014	0.021
C_0	0.987	0.955	0.972	0.999	1.025	0.843	0.994	0.967	0.901	1.036	0.967	0.884	1.030	0.837	0.854	1.005	0.793	0.829	0.967	1.038	0.809
Time-Critical Optimized as Equation 3																				Error: 0.2191	
C	0.000	0.107	0.300	0.000	0.203	0.300	0.000	0.146	0.300	0.000	0.079	0.300	0.000	0.094	0.300	0.000	0.071	0.300	0.000	0.000	0.300
R	0.018	0.025	0.039	0.026	0.040	0.036	0.059	0.006	0.080	0.020	0.005	0.021	0.069	0.008	0.091	0.015	0.005	0.019	0.003	0.100	0.100
C_0	1.001	1.010	0.984	1.002	1.020	0.976	1.000	0.972	0.951	1.003	1.009	0.881	0.999	0.981	0.942	1.001	1.004	0.826	1.000	1.092	0.820
C_1	-0.085	-0.195	-0.067	-0.088	-0.375	-0.129	-0.107	-0.200	-0.158	-0.085	-0.291	-0.006	-0.120	-0.154	-0.337	-0.087	-0.269	0.014	-0.100	-1.000	-0.049

Table 5: Fuzzy Clustering Models of a 2-D Function

References

1. G.J. Klir, T.A. Folger, *Fuzzy Sets, Uncertainty, and Information*, Prentice-Hall, 1988.
2. H.-J. Zimmermann, *Fuzzy Set Theory And Its Applications*, Kluwer Publishers, 1991.
3. T. Takagi, M. Sugeno, "Fuzzy Identification of Systems and Its Application to Modeling and Control," *IEEE Transactions on Systems, Man & Cybernetics*, Vol 15, pp. 116-132, 1985.
4. W.E. Combs, "Optimized Operators for Time-Critical Applications," Unpublished, 1994.
5. R. Zhao, R. Govind, "Defuzzification of Fuzzy Intervals," *Fuzzy Sets and Systems*, Vol 43, No. 1, pp. 45-55, September 1991.
6. H. Hellendoorn, C. Thomas, "Defuzzification in Fuzzy Controllers," *Journal of Intelligent and Fuzzy Systems*, Vol 1, pp. 109-123, John Wiley and Sons, 1993.
7. B. Kosko, "Fuzzy Systems As Universal Approximators," Proceedings of the 1992 IEEE International Conference on Fuzzy Systems, pp. 1153-1162.
8. B. Kosko, *Neural Networks and Fuzzy Systems*, Prentice-Hall, 1992.
9. H.R. Berenji, P.S. Khedkar, "Clustering in Product Space for Fuzzy Inference," Proceedings of the 1993 IEEE International Conference on Fuzzy Systems, pp. 1402-1407.
10. D.E. Hoak, D.E. Ellison, et al, *USAF Stability and Control DATCOM*, Flight Control Division, Air Force Flight Dynamics Laboratory, Wright Patterson Air Force Base, 1968 Edition.

AIAA-97-3790

SIMULATION OF A TAILLESS AIRCRAFT USING NONLINEAR INDICIAL RESPONSE AERODYNAMIC MODELING METHODS

Phillip D. McKeehen* James H. Myatt# Deborah S. Grismer‡ James M. Buffington#
Air Force Research Laboratory(AFRL/FIGC)
Wright-Patterson Air Force Base, Ohio 45433-7531

ABSTRACT

A nonlinear simulation of the high angle-of-attack(AOA) rolling motion of a tailless aircraft with 65 degree sweep and delta-like planform was developed. The simulation used a FORTRAN 77 representation of a Nonlinear Indicial Response (NIR) aerodynamic model of a rolling 65 degree delta wing. The standard linear stability derivative for rolling moment was replaced by an NIR model which accounts for the large time lags associated with the slow movement of the vortex breakdown locations. The NIR model's software was integrated with previously written software simulating the 6-DOF motion of the vehicle. A simple controller that tracks body-axis rate commands was designed using the standard linear aerodynamic model. While the control law provided tight roll-rate tracking for the standard linear aerodynamic model, significant tracking degradation was observed for the NIR aerodynamic model at high angles-of-attack. This tracking degradation indicates that failure to account for the large lags associated with vortex burst point propagation in the control law design can lead to inaccurate simulation of the responses to control inputs.

INTRODUCTION

A major area of current research in the aerospace community is concerned with the analysis and design of vehicles without vertical tails. While these vehicles offer certain advantages over conventional aircraft with tails, they also present considerable challenges to the designer. Lack of a vertical tail forces the flight control system designer to use other yaw control effectors for enabling the aircraft to still possess the requisite maneuverability for combat situations.

Reference [1] describes the development of robust flight control laws and their evaluation using simulation for a tailless fighter design being studied by the Air Force Research Laboratory's Control Dynamics Branch. This design evolved from a research program intended to explore new control effectors such as all-moving tips, differential leading-edge flaps, spoiler-slot deflectors and pitch and yaw thrust vectoring. The control laws described in [1] were designed for a configuration which included all-moving tips for yaw control and pitch and yaw thrust vectoring along with the traditional pitch control surfaces of pitch flaps and elevons on a 65-degree delta wing configuration(see Fig. 1 in [1]). The resulting simulation was implemented in the GENeral Environment for the Simulation of Integrated Systems (GENESIS) software environment. The design evaluation noted in [1] showed acceptable flying qualities, stability and robustness to inherent aircraft modeling uncertainties within the design envelope of Machs ranging from .3 to .5 and altitudes ranging from 10,000ft to 20,000ft.

The aerodynamic model of the vehicle used in the simulation referred to above was based on traditional linear methods for

* Senior Member AIAA
Aerospace Engineer, Member AIAA
‡ Captain, USAF, Member AIAA

This paper is declared a work of the U.S. Government and is not subject to copyright protection in the United States.

estimating stability and control derivatives and using them in the build-up of the total aerodynamic coefficients. Figure 1 shows a plot of experimental data for this vehicle consisting of rolling moment coefficient vs sideslip for an angle of attack of 31.5 degrees. As this plot shows, for a sideslip with absolute value greater than 6 degrees the actual trend is highly nonlinear.

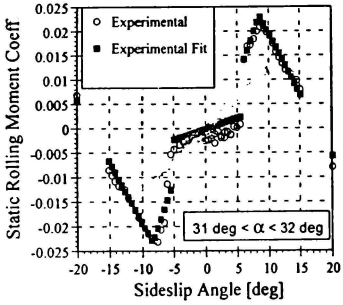


Fig 1 Rolling Moment Coefficient ($\alpha = 31.5°$)

In addition to such nonlinearities exhibited in the aerodynamic behavior of this vehicle at high angles of attack, the vortices associated with flow separation at the delta wing's leading edge are known from experimentation to often burst over the wing, producing highly turbulent flow aft of the breakdown location. In general, the location of this vortex burst point is a function of both angle of attack and sideslip angle. The movement of the breakdown location gives rise to time lags in the flow field relative to disturbances such as the maneuvering motion of the aircraft. The presence of these time lags makes modeling the aerodynamic forces and moments difficult for high-angle-of-attack flight. Furthermore, for vehicle motions that cross a critical state(e.g. vortex bursting crossing the trailing edge or secondary vortices lifting off the wing surface), additional transient effects result and persist long after the critical state has been encountered. [2] demonstrates that these transient effects cause linear and locally linear aerodynamic models to give poor predictions of aerodynamic loads.

This paper describes a research effort concerning the use of Nonlinear Indicial Response(NIR)[3] methods in tailless aircraft simulation. These methods are an alternative to the standard modeling methods such as the (linear) stability and control derivatives. The objective of the research was to determine the effects of multiple time scales, large lags and critical state transient effects on control law performance. The test maneuvers were designed to take advantage of the large database that is available for a 65-deg delta wing from earlier studies.

Specifically, an approximate NIR method [4] was used to provide an alternate expression for the rolling moment coefficient build-up equation at high angles of attack. Recently collected wind tunnel data for the same configuration used in [1], in coordination with the recent modeling applied to the 65-degree delta wing were used to model the time dependent effects on the rolling moment. The simulation's FORTRAN code was modified to implement the NIR methodology, as described in [4] and [5]. The paper describes the nonlinear aircraft simulation, NIR modeling of rolling moment, control law design, the effects of the NIR modeling on the control law's performance and the corresponding implications for the accuracy of the simulation.

The following facts summarize the design and conduct of this research. The narrow scope of the effort should be carefully noted while interpreting the results and their implications for tailless aircraft flight control and simulation:

(1) This is the first instance of the NIR modeling method being implemented in a nonlinear simulation environment. Only the rolling moment is modeled by NIR methods.

(2) It is desirable in this study to allow sufficient sideslip during maneuvers to permit critical state crossings. Therefore, the control law is not designed to directly attenuate sideslip, but only to provide adequate response to high-AOA body-axis rate commands.

(3) The vehicle simulated in this effort has a planform (see Fig 2) similar to that used in [1]. The simulated vehicle is artificially driven by body-axis moments as opposed to control deflections to isolate NIR modeling

effects. It is assumed that the required moment is obtainable.
(4) The aerodynamic data used in the NIR rolling moment model were collected specifically for body-axis rolling motions at high-AOA. Thus the maneuvers used in the simulation were simple body-axis rolls. They are not meant to be representative of realistic tactical aircraft maneuvers.

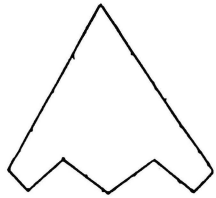


Fig 2 Tailless Aircraft Planform

THE GENESIS ENVIRONMENT

GENESIS is Northrop-Grumman Corporation proprietary simulation software. It is supported on a number of different platforms. GENESIS is capable of performing in real-time or nonreal-time, but was only used in nonreal-time for this research effort. [6] provides a complete discussion of its structure and functionality .

GENESIS is a tool that can be used to simulate any time-varying system and has been used to simulate guided weapons, mass transit vehicles and automobiles. However, since GENESIS evolved from aircraft simulation applications, many standard aircraft features are built in. For example, software modules exist for the 6 DOF rigid body equations of motion, a standard atmosphere model and standard disturbance models. Only vehicle specific FORTRAN models such as actuators, propulsion, airframe, landing gear, and control laws or aerodynamic, propulsion and control law databases must be provided by the user.

Besides the standard components, GENESIS has simulation core software comprised of built-in routines for linear model generation, numerous trimming options, database management and many utility functions (control system filters,matrix manipulations,etc.). The simulation core has another feature that makes it particularly appealing as a simulation tool: an extensive and powerful user interface. Since GENESIS has been developed primarily for interactive use, it incorporates a wide range of user commands that permit a diverse array of analyses to be performed.

VEHICLE MODEL

Standard Aerodynamic Model

A standard aerodynamic model is used for control law development and performance comparisons. The mathematical form of the rolling moment is

$$C_l(t) = C_{l_s}(\alpha(t),\beta(t)) + C_{l_p}p(t) + C_{l_r}r(t) + \delta C_l$$

The static term $C_{l_s}(\alpha(t),\beta(t))$ is a linear interpolation between angle of attack data spaced at 2.5° increments and sideslip data at $\beta = 0°$ and $\pm 10°$. For sideslip angles greater than 10°, the static rolling moment value is held at the $\beta = 10°$ value. This restricts the useful sideslip angle regime to less than approximately 13°. Figure 3 shows the standard aerodynamic model static rolling moment for $\alpha = 29°$, compared to the experi-

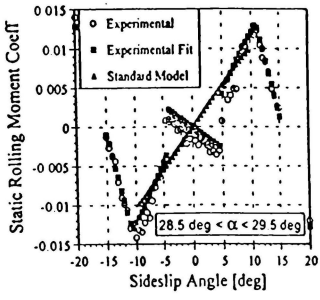


Fig 3 Comparison of Experimental C_{l_s} with Its New Curve Fit and the Standard Representation

mental data obtained at .5 degree increments. The angular rate derivatives C_{l_p} and C_{l_r} are interpolated between angles of attack with typical values: $C_{l_p}(\alpha = 30°) = -.27 \text{rad}^{-1}$ and $C_{l_r}(\alpha = 30°) = .154 \text{ rad}^{-1}$. The final term δC_l is the total rolling moment increment due to control input.

NJR Aerodynamic Model

Static rolling moment

The static rolling moment must be well defined before a model can be developed for unsteady conditions. Figure 3 shows the static rolling moment at an angle of attack of approximately 29° for sideslip angles between $\beta = \pm 15°$. The open circles represent time-averaged experimental values. Critical states, defined as discontinuities in the forces/moments or their slopes, occur at sideslip angles of $\beta = \pm 5°$ and $\pm 10°$. Based on 65-degree delta wing data, the critical state at $\beta = 5°$ is associated with the vortex burst point on the leeward winghalf (in a crossflow sense) jumping across the trailing edge, producing a discontinuity in the rolling moment [7]. Jenkins et al. [8], show that a model of the static data is greatly improved if the motion variables (α, β for the present case) are split into regions bounded by critical states. Figure 4 shows region boundaries for the angles of attack/sideslip angles of interest. The mathematical form for the static rolling moment in each region is assumed to be

$$C_{l_s} = (m_1\alpha + b_1)\beta + (m_0\alpha + b_0) \qquad (1)$$

For a given angle of attack, the modeled rolling moment is a linear function of sideslip angle. The slopes and zero sideslip static rolling moments for each region are themselves linear functions of angle of attack. Table 1, shown below, gives the coefficients for the C_{l_s} relationships for each region.

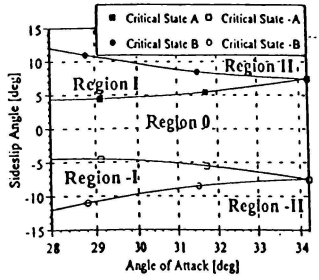


Fig 4 Critical State and Region Locations

Potential and Vortical Contributions

The transient response of the flow field to changes in model attitude contains multiple time scales; these multiple time scales must be accounted for separately to accurately model the rolling moment under dynamic conditions.

Region	m_1	b_1	m_0	b_0
0	3.85×10^{-4}	-1.18×10^{-2}	0	0
I	8.26×10^{-4}	-2.22×10^{-2}	-1.87×10^{-3}	4.95×10^{-2}
-I	8.26×10^{-4}	-2.22×10^{-2}	1.87×10^{-3}	-4.95×10^{-2}
II	1.55×10^{-4}	-7.29×10^{-3}	-9.96×10^{-5}	4.69×10^{-2}
-II	1.55×10^{-4}	-7.29×10^{-3}	9.96×10^{-5}	-4.69×10^{-2}

Table 1 Coefficients for C_{l_s} Relations

Much of the flowfield reacts at the convective time scale, which is considered to be instantaneous for the rates studied here. However, the vortex burst points respond slowly to changes in model attitude and have a time scale that is an order of magnitude slower than the convective time scale. The fast reacting portion of the rolling moment response is assumed to be due entirely to potential effects (no vortices) and the slow reacting portion is assumed to result from the vortical effects.

The fast reacting component of the rolling moment is found using HASC95 (High Angle of Attack Stability and Control), a preliminary design level subsonic aerodynamic prediction code. HASC95 predicts the forces/moments given geometry and flow conditions (Mach, AOA, sideslip, etc.).

To accurately determine the potential

effects alone, a flag is set in the input file so that vortical effects are not included in the calculations. An estimate of the potential contribution to the static rolling moment is given by

$$C_{l_s}^{p}(\alpha,\beta) = A_3\beta^3 + A_1\beta \qquad (2)$$

where $A_3 = -3.97x10^{-8}\alpha + 2.56x10^{-6}$

and

$A_1 = 9.43x10^{-6}\alpha^2 - 5.09x10^{-4}\alpha + 2.25x10^{-3}$

HASC95 is also used to determine the potential flow contribution to roll damping. $C_{l_p}^{p}$ is essentially constant for the angles of attack/sideslip angles of interest with a value of $C_{l_p}^{p}$ = -0.065 rad^{-1}.

The vortical contribution to the static rolling moment is calculated by subtracting the potential contribution from the total static value. The potential and vortical contributions to the static rolling moment for an angle of attack of 29° are shown in Figure 5 .

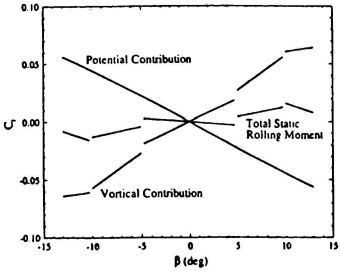


Fig 5 Potential and Vortical Contributions to Static Rolling Moment

Total Rolling Moment

The model of the rolling moment is based on nonlinear indicial response theory and makes the following assumptions:

1) rigid body

2) constant or slowly-varying flight-path properties (speed, air density, etc.)

3) the motion is an analytic function of time

4) the motion starts from static equilibrium

The assumed mathematical form is

$$C_l(t) = C_{l_s}^{v}(\alpha(0),\beta(0)) + C_{l_{s_{lag}}}^{v}(t) + C_{l_s}^{p}(\alpha(t),\beta(t)) + C_{l_{p_{lag}}}^{v}(t) + C_{l_p}^{p}p(t) \qquad (3)$$

The individual components are defined as :

$C_{l_s}^{v}(\alpha(0),\beta(0))$ is the vortical contribution to the static rolling moment at the time the motion is started from equilibrium.

$C_{l_{s_{lag}}}^{v}(t)$ is a lag of the vortical contribution to the static rolling moment.

$C_{l_s}^{p}(\alpha(t),\beta(t))$ is the potential contribution to the static rolling moment. It is assumed to act instantaneously.

$C_{l_{p_{lag}}}^{v}(t)$ is a lag network applied to the roll rate and is associated with the vortical effects.

$C_{l_p}^{p}p(t)$ is the potential flow roll damping; it is also assumed to act instantaneously.

The lag of the vortical contribution to the static rolling moment is found by numerically integrating

$$\frac{dC_{l_{s_{lag}}}^{v}(t)}{dt} = \frac{\left[C_{l_s}^{v}(\alpha(t),\beta(t)) - C_{l_{s_{lag}}}^{v}(t)\right]}{T_c} \qquad (4)$$

Recalling the requirement that the motion start from a static equilibrium condition,

$$C_{l_{s_{lag}}}(0) = C_{l_s}^{v}(\alpha(0),\beta(0))$$

Application of the lag network to the roll rate gives the state-space equations

$$\frac{dx(t)}{dt} = \frac{-x(t)}{T_c} + \frac{a_0 - a_1 / T_c}{T_c} p(t) \qquad (5)$$

$$C^{v}_{l_{p_{lag}}}(t) = x(t) + \frac{a_1}{T_c} p(t) \qquad (6)$$

where *x(t)* is a dummy internal state variable. Again making use of the static equilibrium initial condition assumption, x(0)=0. The rolling moment is calculated by numerically integrating Eqs. 4,5 and 6 using a 4th-order Runge-Kutta method and adding the potential effects and initial conditions as shown in Eq. (3). The parameters for each region, enumerated in Table 2, were estimated using unsteady data for a flat 65-degree delta wing with sharp leading edges and a circular centerbody rolling about the longitudinal body-axis.

Whenever a critical state is crossed, the integrals must be split and a transient effect $\Delta C_l(\alpha(t_{c.s.}),\beta(t_{c.s.});t,t_{c.s.})$ must be added to account for the persistent effects associated with crossing the critical state and to allow the rolling moment to reach the correct static value whenever the motion is arrested. The motion for the original integrals of Eqs. 4,5 and 6 is *frozen* at $(\alpha(t_{c.s.}-\varepsilon),\beta(t_{c.s.}-\varepsilon))$ and the integration is continued to allow the rolling moment due to the motion prior to the critical state crossing to reach its static value $C_{l_s}(\alpha(t_{c.s.}-\varepsilon),\beta(t_{c.s.}-\varepsilon))$. A second set of integrals of Eqns. 4,5 and 6 are started from a static equilibrium condition at $(\alpha(t_{c.s.}+\varepsilon),\beta(t_{c.s.}+\varepsilon))$. Then, the transient effect $\Delta C_l(\alpha(t_{c.s.}),\beta(t_{c.s.});t,t_{c.s.})$ is added to complete the prediction of the rolling moment. The transient effects are discussed in detail in [9]. Note that the control effects are not included in the NIR model. While they are most assuredly important and may affect both the lags and the critical state locations, no data exist at this time to determine the dynamic response to control inputs.

Region	T_c, t'	a_1/T_c	a_0
0	31.0	0.55	2.84
I,-I	28.5	0.62	1.98
II,-II	0.5	0.5	0.5

Table 2 Model Parameters

Engine model

The simulation uses a model of the F-110-GE-229 engine. The engine thrust is a function of power level angle(PLA), Mach number and altitude. The NIR model requires that rolling motion be initiated from a static equilibrium condition. Thus, the aircraft model was necessarily trimmed at approximately 30 degrees AOA before roll commands were issued. This required thrust trim values beyond maximum thrust levels of the F-110-GE-229 engine. Tested flight conditions required artificially extrapolated thrust trim values between approximately 13,000 and 16,000 lbs. Simulation runs yielded thrust values between approximately 12,000 and 17,000 lbs.

CONTROL LAW DESIGN

The flight control system in this study stabilizes the short-period and dutch-roll modes and provides decoupled first-order response to body-axis rate commands. Note that the controller was not designed to attenuate sideslip. The flight controller is based upon feedback linearization [10] or dynamic inversion [11,12] control theory which is a model-based approach. Since aero- dynamic modeling is inexact, model-based controllers must also be robust to unmodeled dynamics and plant parameter uncertainty. A limited analytical study [13] and many applied efforts [14,12,1] have shown dynamic inversion to be a robust flight control design technique.

The rigid-body equations of motion are well documented [15,16] and are approximated well by the following form

$$\begin{bmatrix} \dot{z} \\ \dot{y} \end{bmatrix} = \begin{bmatrix} f_z(z,y) \\ f_y(z,y) \end{bmatrix} + \begin{bmatrix} g_z(z,y) \\ g_y(z,y) \end{bmatrix} u \qquad (7)$$

where z are the uncommanded aircraft states, y are the commanded aircraft states, and u are the control effectors. The commanded states are chosen to be the body-axis rates

$$y = [P \quad Q \quad R]^T \tag{8}$$

where P is the body-axis roll rate, Q is the body-axis pitch rate, and R is the body-axis yaw rate. Body-axis rate commands are desirable at the high angle-of-attack flight conditions of this study. Further, the rigid-body short-period, roll subsidence and Dutch roll flight control modes are augmented well with body-axis rate feedback. NIR modeling only affects stability derivatives contained in f_z and f_y from eq. (1). Therefore, NIR modeling effects are isolated by removing the control derivative effects, i.e. g_z and g_y. The tailless aircraft configuration in this study has multiple innovative and conventional control effectors to overcome multi-axis instabilities. All of the control effectors, however, primarily produce body-axis moments so $g_z(z,y) \approx 0$. The effects of g_y are artificially removed by defining the reduced dimension control vector $d_y \equiv g_y(z,y)u$ which results in the following system

$$\begin{bmatrix} \dot{z} \\ \dot{y} \end{bmatrix} = \begin{bmatrix} f_z(z,y) \\ f_y(z,y) \end{bmatrix} + \begin{bmatrix} 0 \\ d_y \end{bmatrix} \tag{9}$$

The dynamic inversion feedback control law is given by:

$$d_y^{cmd} = -\hat{f}_y(\hat{z},\hat{y}) + \omega_c(x_i - \hat{y}) + \omega_c f_c y^{cmd} \tag{10}$$

$$\dot{x}_i = -\omega_c^2 f_i(\hat{y} - y^{cmd})$$

$$\omega_c = 5 \ \frac{rad}{\text{sec}}$$

$$f_i = .25$$

$$f_c = .5$$

where $\hat{z}$ and $\hat{y}$ are state measurements, ω_c is a proportional control gain, f_i is an integral control gain, f_c is a feedforward command gain, and $\hat{f}_y$ is an estimate of f_y. The feedforward gain provides first order command response in all axes, and the feedback gains provide stability and robustness to unstructured uncertainty levels consistent with fighter aircraft [14]. Note that f_y contains aerodynamic, propulsive, and inertia coupling components. The estimate $\hat{f}_y$ contains inertia coupling terms with constant moments of inertia, and terms generated using least-squares estimates of the aerodynamic and propulsive coefficients [12].

SIMULATION RESULTS

The vehicle's response to high angle of attack body axis roll rate commands was simulated using GENESIS. Simulations using the standard linear stability derivative for rolling moment and the NIR rolling moment model were performed for roll rate commands of 10 deg/sec and 20 deg/sec. The aircraft was trimmed at an angle of attack of either 29.1° or 31.5° before issuing the commands. These commands were held for a length of time necessary to produce either 0, 1 or 2 critical state crossings and then a zero deg/sec command was issued. Recall, as stated in the introduction, that sideslip is not directly attenuated. The vehicle was simply commanded to roll for a duration of time at a rate to produce the number of critical state crossings associated with the test cases defined below. The time histories in Fig 6 - 17 correspond to these 6 cases:

Case	α_{trim} (degrees)	Roll Rate Comm. (deg/sec)	Critical States Crossed
1	31.5	10	0
2	31.5	20	0
3	29.1	10	1
4	29.1	20	1
5	31.5	10	2
6	31.5	20	2

Table 3 Simulation Cases

For each case, the range of angles of attack attained is noted and the roll rate response to the command and corresponding sideslip time-histories for Standard and NIR rolling moment models are graphed.

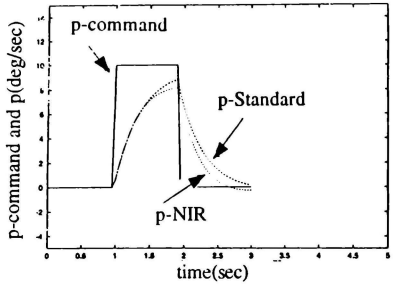


Fig 6 Case 1 Roll Rate Command and Response ($31.4^\circ \le \alpha \le 31.7^\circ$)

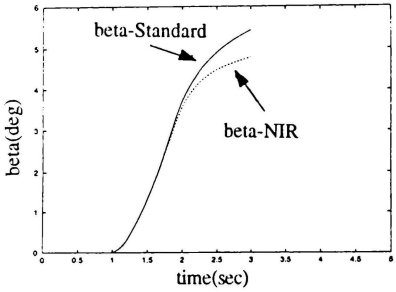


Fig 7 Case 1 β Response ($31.4^\circ \le \alpha \le 31.7^\circ$)

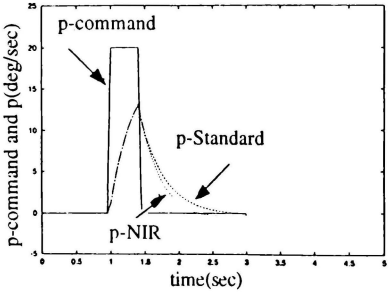


Fig 8 Case 2 Roll Rate Command and Response ($31.4^\circ \le \alpha \le 31.8^\circ$)

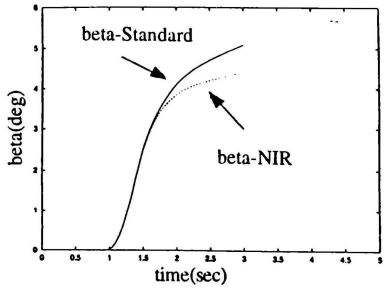


Fig 9 Case 2 β Response ($31.4^\circ \le \alpha \le 31.8^\circ$)

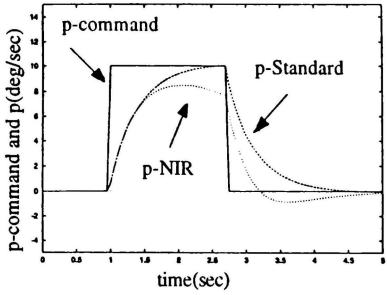


Fig 10 Case 3 Roll Rate Command and Response ($28.6^\circ \le \alpha \le 29.7^\circ$)

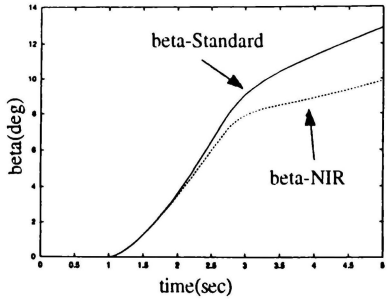


Fig11 Case 3 β Response ($28.6^\circ \le \alpha \le 29.7^\circ$)

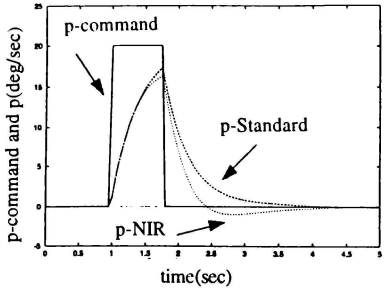


Fig 12 Case 4 Roll Rate Command and Response ($28.6^\circ \le \alpha \le 29.9^\circ$)

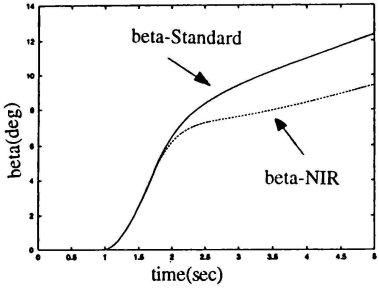


Fig13 Case 4 β Response ($28.6^\circ \le \alpha \le 29.9^\circ$)

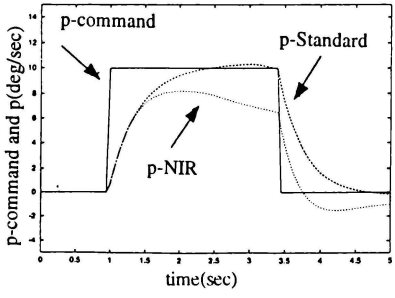


Fig 14 Case 5 Roll Rate Command and Response ($29.8^\circ \le \alpha \le 31.7^\circ$)

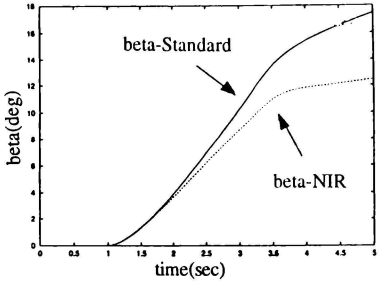


Fig15 Case 5 β Response ($29.8^\circ \le \alpha \le 31.7^\circ$)

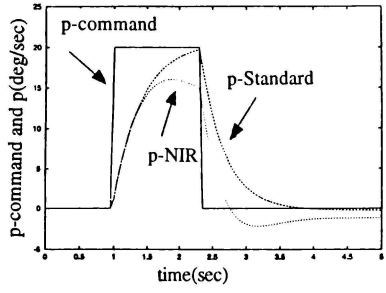


Fig 16 Case 6 Roll Rate Command and Response ($29.4^\circ \le \alpha \le 31.7^\circ$)

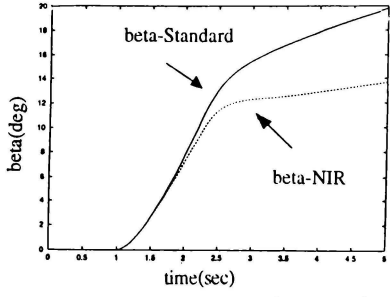


Fig17 Case 6 β Response ($29.4^\circ \le \alpha \le 31.7^\circ$)

A close examination of Figures 6 through 17 reveals several interesting trends. Note that the control law design was known to yield a first order response [13] to a step command in roll rate using the standard rolling moment model. Such responses have no overshoot. Figures 6 through 9 correspond to no critical state crossings. Here no significant degradation of the roll rate responses is seen when the standard model is replaced by the NIR model. Sideslip (β) responses for both 10 deg/sec and 20 deg/sec commands are similar for NIR and standard models.

Figures 10 through 13 correspond to 1 critical state crossing. Here β responses show similar patterns for the 10 and 20 deg/sec commands, but a growing difference in magnitude between NIR and standard model responses after the issuance of the 0 deg/sec command. There is significant degradation in the roll rate response when the NIR model is used in a command with 1 critical state crossing. In Figure 12 significant overshoot occurs in the 0 deg/sec command's response at t = 2.8 sec, although there is no significant degradation of the 20 deg/sec command's response. Note that for the 10 deg/sec command not only is there overshoot in the NIR response at t = 3.5 sec, but there is significant degradation in the response between t=1.5 and t=2.7 sec. Indeed, the NIR response slows noticeably starting at t = 1.5 sec and is actually reversing directions at t = 2.3 sec.

For 2 critical state crossings (Fig 14 - 17) there is significant degradation in the roll rate response for the NIR model for both the 10 deg/sec and 20 deg/sec commands. For these cases, not only does the NIR response significantly overshoot the 0 deg/sec commands, but there is highly noticeable slowing and reversing of the response to the 10 and 20 deg/sec commands.

The analysis of these time histories indicates the following:

(1) Replacing the standard linear stability derivative rolling moment model with an NIR model can cause significant degradation in a tailless aircraft's roll rate response to high-AOA body axis roll rate commands.

(2) The degradation is aggravated as the motion is modified to allow the crossing of an increasing number of critical states.

CONCLUSIONS

A nonlinear simulation of the high-AOA rolling motion of a tailless aircraft with 65 degree sweep and delta-like planform was developed. The simulation used a FORTRAN 77 representation of a Nonlinear Indicial Response (NIR) aerodynamic model of a rolling 65 degree delta wing. The standard linear stability derivative for rolling moment was replaced by an NIR model which accounts for the large time lags associated with the slow movement of the vortex breakdown locations.

A simple control law was implemented to provide decoupled body-axis rate command response at high angles of attack. In order to permit critical state encounters, the control law was not designed to directly attenuate sideslip.

A comparison was made of the closed-loop tracking performance when the standard rolling moment model was replaced by the NIR model. The vehicle's response to 10 deg/sec and 20 deg/sec body axis roll rate commands at high-AOA was simulated. Results indicate that failure to account for the large lags associated with vortex burst point propagation in the simulations at high-AOA can lead to an inaccurate prediction of the responses to control inputs. Furthermore, the inaccuracy is aggravated as the motion is modified to allow the crossing of an increasing number of critical states. Finally, given the degradation in performance, the large lags should be accounted for in the design of the control law.

ACKNOWLEDGMENTS

The authors would like to acknowledge WL Emeritus researcher Mr Jerry Jenkins for the technical insight he provided in this research effort. They also wish to acknowledge Mr William Blake for his HASC95 support and for first suggesting the research idea of using the NIR model in a simulation of this particular tailless configuration's high-AOA rolling motion.

REFERENCES

1. Ngo, A.N., Reigelsperger, W.C., Banda, S.S., Bessolo, J.A., "Multivariable Control Law Design for a Tailless Aircraft", 1996 AIAA Guidance, Navigation, and Control Conference, AIAA 96-3866, July 1996.

2. Jenkins, J.E., Myatt, J.H., Hanff, E.S., "Body-Axis Rolling Motion Critical States for a 65-Degree Delta Wing", AIAA Paper 93-0621, Jan 1993.

3. Tobak, M., Chapman, G.T., Schiff, L.B., "Mathematical Modeling of the Aerodynamic Characteristics in Flight Mechanics", Proc of the Berkeley-Ames Conf on Nonlinear Problems in Control and Fluid Dynamics, Math Sci Press, Brookline, MA, 1985, pp. 435-450; also NASA TM-85880, Jan 1984.

4. Myatt, J.H., "Modeling the Rolling Moment on the 65-Degree Delta Wing for Rolling Motions at High Angle of Attack", Ph. D. Thesis, Stanford University, 1997.

5. Myatt, J.H., Jenkins,J.E., "A Nonlinear Indicial Response Model for The Rolling 65-Degree Delta Wing", Proc. AIAA Atmospheric Flight Mechanics Conf., San Diego, CA, July 1996.

6. Wong,J.P. et al., "Companion Guide to GENESIS Software Version 2.2", Vol I - User's Guide, Vol II - Programmer's Guide, Vol III - Linearization Guide, Northrop Corporation B-2 Division, September 1993.

7. Jobe, C. E., Hsia, A. H., Jenkins, J. E., and Addington, G. A., "Critical States and Flow Structure on a 65-Degree Delta Wing", Journal of Aircraft, Vol. 33, No. 2, 1996, pp. 347-352.

8. Jenkins, J. E., Myatt, J. H., and Hanff, E. S., "Body-Axis Rolling Motion Critical States of a 65-Degree Delta Wing", Journal of Aircraft, Vol. 33, No. 2, 1996, pp. 268-278.

9. Grismer, D.S., Myatt, J.H., McKeehen, P.D., Buffington, J.M., "Application of a Nonlinear-Indicial-Response Model to a Tailless Aircraft Simulation", AIAA Paper 97-3644, Atmospheric Flight Mechanics Conference, New Orleans, LA, August 1997.

10. Isidori, A., *Nonlinear Control Systems,* Springer-Verlag, 1989.

11. Enns, D., Bugajski, D., Hendrick, R., and Stein, G., "Dynamic inversion: an evolving methodology for flight control design", *International Journal of Control* Vol. 59, No. 1, 1994, pp. 71-91.

12. Honeywell Technology Center, "Application of Multivariable Control Theory to Aircraft Control Laws, Final Report: Multivariable Control Design Guidelines", WL-TR-96-3099, May 1996.

13. Morton, B., Enns, D., Zhang, B.Y., "Stability of Dynamic Inversion Control Laws Applied to Nonlinear Aircraft Pitch Axis Models", IMA Preprint Series # 1245, July 1994.

14. Buffington, J.M., "Control Design and Analysis for Systems with Redundant Limited Controls", Ph. D. Thesis, University of Minnesota, 1996.

15. Miele, A., *Flight Mechanics: Theory of Flight Paths, Vol 1,* Addison-Wesley, Reading, MA, 1962.

16. McRuer, D., Ashkenas, I., Graham, D., *Aircraft Dynamics and Automatic Control,* Princeton University Press, Princeton , NJ, 1973.

AEROELASTIC MODELING OF MACROBODY AIRCRAFT FOR REAL-TIME-FLIGHTSIMULATION

M. Dahms*, W. Kindel*, R. Liebich**, O. Schmidt*
Aerospace Institute
Berlin University of Technology
Marchstraße 12
D-10587 Berlin, Germany

The presented project was founded by the German Minister of Education, Science, Research and Technology under the project number 20 A 9503C. The authors are responsible for the content.

Abstract

The trend to larger transport aircraft leads to new problems, which require an integrated point of view already in the pre-development phase. Due to the close coupling of flight mechanical and structural modes a quasi-stationary model is no longer adequate to describe the aircraft behavior for flightmechanical investigations on flying qualities.

For some of those investigations pilot-in-the-loop analysis of flight dynamics are necessary, which have to be carried out using a flight simulator. This paper presents the development of a real-time capable model for a very large, flexible transport aircraft.

1. Introduction

At present concept studies are carried out for a new aircraft with a capacity of 600-800 passengers. A large number of new problems arise that have to be solved within the development of such aircraft (here called Macrobody).

Structural stiffness cannot be raised in the same proportion as the aircraft dimensions. Thus the enlargement of aircraft dimensions leads to increasing aeroelastic deflections. Furthermore, existing airport facilities and infrastructure limit the maximum sizes. Both reasons lead to a maximum wing span and length of 80 m (262 ft) for such a Macrobody. Fig. 1 shows the dimensions of a design study with a wing span of 77 m (253 ft) and a length of 75 m (246 ft) at a MTOW of about 550 t (in comparison with an Airbus A340).

Especially two major problems are important from the flightmechanical point of view:

- The lowest structural frequencies of such a Macrobody are within the 1 Hz frequency range. Therefore considerable coupling effects with the flightmechanical degrees of freedom and the Electronic Flight Control System are to be expected.
- The elastic deflections, in particular wing distorsion, cause an immensely reduced effectiveness of the conventional aerodynamic control surfaces.

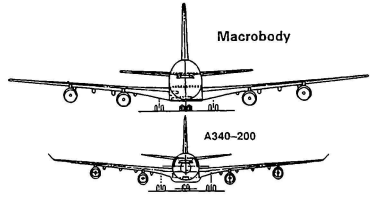


Fig.1 Macrobody and Airbus A340 in comparison

Both facts have an important influence on the flying qualities of such Macrobodies. Therefore the design process must regard these effects from the very beginning.

The investigation on Macrobody flying qualities is the focus of a research-project at the Aerospace Institute of the Technical University of Berlin ("Flying qualities of very large, flexible transport aircraft"). The objective is to improve knowledge in steering and control of such aircraft and to prepare a flying quality database on account of a detailed sensitivity analysis. This data base can be used as an information basis for finding an optimal compromise between structural loads, flying qualities and control effort.

2. Modeling Demands

2.1 The Flightmechanical Model

For investigations of flight dynamics the aircraft is usually assumed to be a single rigid body. This assumption is normally sufficient for conventional aircraft. With larger transport aircraft the structural deflections are covered by additional dynamic pressure dependent coefficients in the aerodynamic model, but structure is not really assigned additional degrees of freedom.

The frequency-levels of the first structural modes of modern aircraft are higher than the flightmechanical modes and, in addition, they are usually well damped at low flight velocities by the aeroelastic feedback. So this simplistic view leads to good results.

Increasing aircraft dimensions decreases structural frequencies in combination with increased aeroelastic deflections. This influences both static and dynamic aircraft behavior. The comparable low structural frequencies of a Macrobody will probably cause considerable coupling effects with the flightmechanical modes. Therefore covering elasticity by quasi-stationary considerations is not applicable and the dynamical effects of flexibility cannot be neglected.

The mathematical model of a Macrobody has to represent the flightmechanical modes as well as the aeroelastic effects in their dynamical interaction in the most realistic way. Only those aeroelastic effects are included that have influence on the flightmechanical behavior.

The necessity to consider aeroelastic effects results in a considerable increase of modeling expenditure because of the determination of dynamical deformations and aerodynamical feedback additionally to the center-of-gravity motion of the aircraft (flightmechanical degrees of freedom). Model development is separated into two fields:

- structure and
- aerodynamics.

Both models are coupled, the aerodynamics supply the most substantial part of the structural loads and the structure defines the rate and condition of deformation, which is decisive for the aerodynamic load distribution.

2.2 The Airbus A340-Full Flight Simulator

An essential basis for flying quality investigations is the assessment by pilots. For that purpose investigations on a flight simulator are necessary. The flight simulator has to reproduce all (important for the assessment) properties of the aircraft in a most realistic way. Therefore the simulator has to be equipped with a modern cockpit with programmable displays, a high quality wide angle visual system as well as a 6-axis-motion system. For the planned tests in the research program the Airbus A330/A340-Full-Flight-Simulator (FFS) of the Center for Flight Simulation Berlin (Zentrum für Flugsimulation Berlin - ZFB) will be used. This simulator is used for pilot training and with an admission corresponding FAA level D fulfills the highest demands.

The flight simulator is extended with a special research component (Scientific Research Facility - SRF), which allows user specific modifications in the real-time software and the integration of user specific models in the simulation process. Fig. 2 shows the principle structure of the A330/A340-Simulator with the SRF. This simulator is the technical basis for the planned tests and defines the marginal conditions relating to hard- and software.

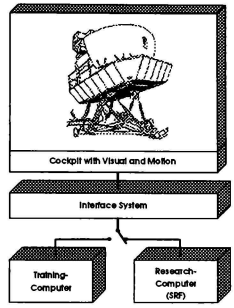


Fig. 2 A340-Simulator with Scientific Research Facility

For the investigations the existing simulation model of the FFS has to be replaced by a new model for a Macrobody, which has to meet the following main requirements:

- Realistic description of the flight dynamical behavior including the aeroelastic effects.
- Adaptability to different project variants.
- Possibility to implement different flight control concepts.
- Real-time capability and compatibility with the simulation environment.

Especially the aspect of real-time capability means a difficult demand and limits the practicable mathematical expenditure. Therefore simple as possible model assumptions are needed, but which have to reproduce the physical facts sufficiently. The given simulation frequency of 60 Hz means that a complete simulation time step has to be executed within 16.6 ms.

The existing simulation model of the A340, satisfying the high demands for pilot training and thus very

complex, was replaced by a new, rather simplified basic structure (Modular Simulation Software - MOSIS)[1]. This software now represents the interface with any flightmechanical model and forms the basis for the model-development of flexible aircraft.

3. Aerodynamic model of the wing

3.1 Stationary Aerodynamic, Lifting Surface Method

The task of the wing aerodynamic model is to calculate the aerodynamic load distribution for a given geometry with known angle of attack distribution. Flexible deformations result in changes of the angle of attack distribution. The model should consider compression and instationary effects the best as possible and should include the influence of control surfaces as well. Therefore the demands to the aerodynamic model are extremely high, especially with regard to real-time requirements.

Lift and Pitching Moment

The lifting surface theory of PRANDTL is the basis of the presented method. According to this theory the wing is replaced by a system of M elementary wings, each having a constant circulation in the spanwise direction. The circulation distribution in wing chord direction is modeled according to BIRNBAUM's first and second distribution functions. Therefore the circulation distribution in wing chord direction can be described by two parameters, which can be obtained by fulfilling the kinematics flow condition at two points along the profile. TRUCKENBRODT chooses the quarter chord line and the wing trailing edge. As a result we get an equation system, which can be written in a matrix form:

$$\begin{bmatrix} \underline{\alpha}_{25} \\ \underline{\alpha}_{100} \end{bmatrix} = \underline{\underline{K}} \begin{bmatrix} \underline{\gamma} \\ \underline{\mu} \end{bmatrix}$$

$\underline{\alpha}_{25}$: angle of attack vector at 25%-line, dimension M
$\underline{\alpha}_{100}$: angle of attack vector at wing trailing edge
$\underline{\gamma}$: local circulation vector, dimension M
$\underline{\mu}$: local pitching moment vector, dimension M
$\underline{\underline{K}}$: matrix, dimension 2M x 2M

The matrix $\underline{\underline{K}}$ is only dependent on the number of elementary wings M and the geometry of the wing. Therefore $\underline{\underline{K}}$ is constant and can be inverted. Thus the circulation distribution $\underline{\gamma}$ and the pitching moment distribution $\underline{\mu}$ can be directly computed from the angle of attack distribution $\underline{\alpha}$. The local coefficients for lift and pitching moment relative to the 25%-line can be calculated from

$$c_{Lv} = \frac{2b}{\ell_v}\gamma_v \qquad c_{m25v} = \frac{2b}{\ell_v}\mu_v$$

b : wing span
ℓ_v : local aerodynamic chord

If these relations are also taken into account in the inverted matrix the local coefficients for lift and pitching moment can be calculated directly from the angle of attack distribution along the 25%-line and wing trailing edge

$$\begin{bmatrix} \underline{c}_a \\ \underline{c}_{m25} \end{bmatrix} = \underline{\underline{K}}_1 \begin{bmatrix} \underline{\alpha}_{25} \\ \underline{\alpha}_{100} \end{bmatrix}$$

This simple matrix equation is real-time suitable and gives the possibility to calculate lift and pitching moment distribution for any angle of attack distribution, which also includes flexible deformations.

Main disadvantages of this method are neglected friction effects, no consideration of side slip angles and the restriction to pure wings or horizontal tail planes - the influence of fuselage and engines are not considered. As a result the shape of the distributions of coefficients along the wing span as well as the total coefficients are faulty. Especially the shape of the distributions has an essential influence on the flexible deformations and must be reproduced as best as possible. To compensate these disadvantages both the shape of distributions and the total coefficients calculated by the lifting surface method (LSM) have to be calibrated. The shape is corrected using a panel method (PM) and the total coefficients are calibrated using a nonlinear data basis for the rigid aircraft.

Correction of the shape of distributions

The panel-method is used to model the aerodynamics of the whole aircraft up to Mach number Ma = 0,7. As this method is not real-time capable it is used to pre-calculate the aerodynamic loads of the rigid aircraft (flying shape) for a certain number of flight conditions (Mach number, angle of attack and angle of side slip)[2]. The LSM is used for the same conditions. The distributions of both methods are compared to pre-calculate correction values which a stored in look-up tables to be used in the real-time simulation[3].

In the real-time simulation loop the distributions of lift and pitching moment coefficients are calculated by the LSM for the rigid wing (flying shape, Index "ref") as well as for the flexible wing (actual shape, Index "flex"). For this the distributions of angle of attack at the 25%-line and at the wing trailing edge have to be determined for both cases.

$$\underline{\alpha}_{ref} = \alpha_{CG} + \Delta\underline{\alpha}_{geo} + \Delta\underline{\alpha}_{c}$$

$$\underline{\alpha}_{flex} = \underline{\alpha}_{ref} + \Delta\underline{\alpha}_{flex}$$

α_{CG} : angle of attack at CG
$\Delta\underline{\alpha}_{geo}$: additional angle of attack vector due to rigid geometry (flying shape)
$\Delta\underline{\alpha}_{c}$: additional angle of attack vector due to rigid body rotations and aileron deflections
$\Delta\underline{\alpha}_{flex}$: additional angle of attack vector due to flexibility

The distributions of coefficients are calculated from:

$$\begin{bmatrix} \underline{c}_L \\ \underline{c}_m \end{bmatrix}_{ref} = \underline{\underline{K}}_1 \begin{bmatrix} \underline{\alpha}_{25} \\ \underline{\alpha}_{100} \end{bmatrix}_{ref} \qquad \begin{bmatrix} \underline{c}_L \\ \underline{c}_m \end{bmatrix}_{flex} = \underline{\underline{K}}_1 \begin{bmatrix} \underline{\alpha}_{25} \\ \underline{\alpha}_{100} \end{bmatrix}_{flex}$$

For the actual flight conditions the calibration values are determined by interpolation from the above mentioned look-up tables. The corrected distributions can be calculated from:

$$\begin{bmatrix} \underline{c}_{L,ref} \\ \underline{c}_{m,ref} \end{bmatrix}_{kor} = \left(\begin{bmatrix} \underline{c}_L \\ \underline{c}_m \end{bmatrix}_{ref} - \begin{bmatrix} \underline{B}_L \\ \underline{B}_m \end{bmatrix} \right) \cdot \begin{bmatrix} \underline{F}_L \\ \underline{F}_m \end{bmatrix} + \begin{bmatrix} \underline{B}_L \\ \underline{B}_m \end{bmatrix}$$

$$\begin{bmatrix} \underline{c}_{L,flex} \\ \underline{c}_{m,flex} \end{bmatrix}_{kor} = \left(\begin{bmatrix} \underline{c}_L \\ \underline{c}_m \end{bmatrix}_{flex} - \begin{bmatrix} \underline{B}_L \\ \underline{B}_m \end{bmatrix} \right) \cdot \begin{bmatrix} \underline{F}_L \\ \underline{F}_m \end{bmatrix} + \begin{bmatrix} \underline{B}_L \\ \underline{B}_m \end{bmatrix}$$

$B_L F_L$: calibration values for the lift coefficient
$B_m F_m$: calibration values for the pitching moment coefficient.

Thus we get distributions of coefficients, which shape is similar to the real wing, but where the total coefficients are still divergent from the real values.

Calibration of the total coefficients

To compensate the neglected friction and boundary layer effects, the total coefficients resulted from the corrected LSM are calibrated using a nonlinear data basis of the rigid aircraft. The total coefficients of the rigid wing are calculated from the corrected coefficient distributions of the rigid wing:

$$C_{L,ref} = \frac{1}{S} \sum_{\nu=1}^{M} c_{L,ref,kor,\nu} \cdot \ell_\nu \cdot \Delta y_\nu$$

$$C_{m,ref} = \frac{1}{S \cdot \ell_\mu} \sum_{\nu=1}^{M} c_{m,ref,kor,\nu} \cdot \ell_\nu^2 \cdot \Delta y_\nu$$

S : wing area
Δy_ν : local span of elementary wing
ℓ_μ : mean aerodynamic chord

These total coefficients are compared to the results of the nonlinear data basis to calculate the relating calibration factors:

$$K_{CL} = \frac{C_{L,tab}}{C_{L,ref}} \qquad K_{Cm} = \frac{C_{m,tab}}{C_{m,ref}}$$

Finally the actual coefficient distributions of the flexible wing are calculated from:

$$\begin{bmatrix} \underline{c}_{L,ges,kor} \\ \underline{c}_{m,ges,kor} \end{bmatrix}_{kal} = \begin{bmatrix} \underline{c}_{L,ges} \\ \underline{c}_{m,ges} \end{bmatrix}_{kor} \cdot \begin{bmatrix} K_{CL} \\ K_{Cm} \end{bmatrix}$$

Induced Drag

According to TRUCKENBRODT the induced drag can also be calculated from the lifting surface theory. First the induced angle of attack α_i is calculated at every of the M sections; in matrix form we get

$$\underline{\alpha}_i = \underline{\underline{K}}_2 \underline{c}_a$$

The matrix $\underline{\underline{K}}_2$ is of the dimension M x M and like $\underline{\underline{K}}_1$ only dependent from the geometry of the wing and the number of profiles M. Therefore $\underline{\underline{K}}_2$ can also be pre-computed. With the definition

$$\underline{\underline{c}}_L = \begin{bmatrix} c_{L1} & 0 & & 0 \\ 0 & c_{L2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & c_{LM} \end{bmatrix}$$

the induced drag can be calculated from

$$\underline{c}_{Di} = \underline{\underline{c}}_L \underline{\alpha}_i$$

respectively

$$\underline{c}_{Di} = \underline{\underline{c}}_L \underline{\underline{K}}_2 \underline{c}_L$$

As the lift coefficient distribution is already calibrated an additional correction of the induced drag coefficient is unnecessary

3.2 Instationary Aerodynamic

The considered frequencies of the structural modes are in the range from 0.7 Hz to 4 Hz. The reduced frequency

$$\omega^* = \omega \, \ell_\mu \, / \, V$$

as a parameter to characterize the influence of instationary aerodynamic effects during cruise is in the range from 0.2 to 1.2. According to FÖRSCHING for $\omega^* < 0.1$ the assumption of a quasi-stationary flow leads to good results. As in our case ω^* is significantly higher, instationary effects have to be taken into account. In contrast to usually used methods for instationary aerodynamics a description in the time domain is necessary.

As the above presented lifting surface method is very suitable for real-time purposes, the consideration of instationary effects should be a modification respectively an extension of this method. The new method is using filter functions for each element of the matrix $\underline{\underline{K}}_1$ of the lifting surface method. These filter functions are determined by using a doublet lattice method (DLM) as a reference, which was placed at the institute's disposal by the German Aerospace Research Establishment (DLR) Göttingen.

For this purpose the LSM has to be modified in such a way that the two angles of attack in one section represent a bending (vertical movement) respectively

torsion along the wingspan axis. This can be easily obtained by taking α_{100} as quantity for bending (index "B") and α_{100} - α_{25} as quantity for torsion (index "T"). Thus we get the following modified representation of the LSM:

$$\begin{bmatrix} \underline{c}_L \\ \underline{c}_{m25} \end{bmatrix} = \underline{K}_1' \begin{bmatrix} \underline{\alpha}_B \\ \underline{\alpha}_T \end{bmatrix}$$

$$K_{ij}' = K_{i,j} + K_{i,M+j} \qquad i = 1...2M,\ j = 1...M$$

$$\alpha_B = \alpha_{100}$$

$$\alpha_T = \alpha_{100} - \alpha_{25}$$

Both for LSM and DLM section by section is stimulated with a certain angle of attack α_B, using the DLM. This is done for N frequencies. From the DLM we get complex results for lift and pitching moment $\hat{c}_a$ and $\hat{c}_m$. Comparing the results of both methods we get for every stimulated section (i=1...M) a set of N discrete values $F_{ij}(\omega_k)$ of the searched transfer functions at each of the influenced sections (j=1...M). The same is done for α_T. These transfer functions describe the influence of instationary effects on lift and pitching moment.

$$F_{ij}(\omega_k) = \frac{\hat{c}_{a\,ij}(\omega_k)}{c_{a\,ij}}$$

i : inducing section
j : influenced section

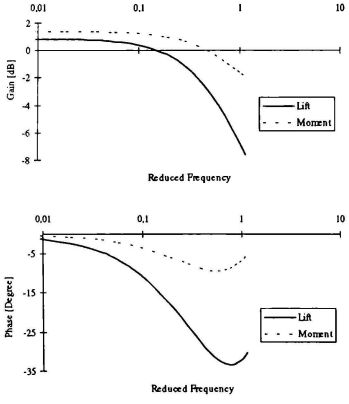


Fig. 3 Influence of section 8 on section 15 [4]

The transfer functions in the range of the interesting frequency range can be approximated by polynomial equations, using an optimization procedure. This has to be done for each element of the matrix $\underline{K}_1$, which means a number of 2M x 2M. With a typical value of M = 31 we get 3844 transfer functions. Fig. 3 shows as an typical example the transfer function for the instationary effect in the influence of section 8 on section 15 in amplitude and phase for $\hat{c}_a$ and $\hat{c}_m$.

As all transfer functions have a common physical reason it is supposed that they can be formulated by one type of function. A PD_2T_2 filter was chosen, whose coefficients were determined by an optimization process.

$$F(s) = K\frac{b_0 + b_1 s + s^2}{a_0 + a_1 s + s^2}$$

Using all 3844 transfer functions means a very high numerical expenditure. This can be reduced by neglecting the transfer functions for those elements, where the stationary contribution at one section is less than 5% of the influence of one section on itself (which gives the maximum at this location). For this example 1246 transfer functions were left, which means a reduction of 68%. Fig. 4 shows the distribution of transfer functions in the matrix $\underline{K}_1'$, each cross indicates a filter to be used.

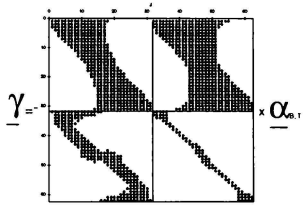


Fig. 4 Filter distribution in the equation system [4]

Obviously there is a systematic in this distribution. Analyzing the structure of the matrix shows that the matrix can be subdivided in four regions:

$$\underline{K}' = \begin{bmatrix} I & II \\ III & IV \end{bmatrix}$$

I : influence of bending on lift
II : influence of torsion on lift
III : influence of bending on pitching moment
IV : influence of torsion on pitching moment

The distribution shown in Fig. 4 shows clearly a dominant influence in the region of the main diagonals of the four sections, because the main diagonals represent the influence of one wing section on itself.

To verify the accuracy of the filter method several results were compared to those of the DLM. Fig. 5 shows the results of both method for a wing bending movement. The figure shows amplitude and phase of lift versus frequency. For comparison reasons the stationary results are included. Up to a reduced frequency of $\omega^* \approx 0.8$ (which means a frequency of $f \approx 2.7$ Hz during cruise) there is a very good corre-

spondence between filter method and DLM. Increasing frequencies lead to increasing errors, especially in phase. Therefore the filter method is a very suitable method to consider instationary effects in real-time applications.

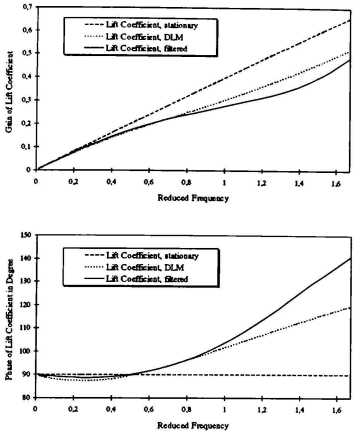


Fig. 5 Lift coefficient for bending movement [4]

4. Structural Model

The basic proceeding of considering the structural elasticity will be shown in the following chapter. It is well known that the elasticity has its main influence on the dynamic wing deflections. The vibrations of fuselage and rudder are not considered in this model because of there small dynamic deformations. Here, the structural model is considering elastic deformations due to flapping, hunting and torsional vibrations.

Because of real-time demands in flightsimulation, the deflections of wings have to be described by a small number of degrees of freedom (DOF). In Structure Dynamics a lot of useful methods for discretizing continuos systems are known. Finite Element is one of these methods. It calculates nearly exact static and dynamic deflections. But Finite Element needs too many DOF's for a correct system description and that means too long computing times. The real-time demand obliges to a much simpler description of the wing model. In a first approach the wing is modeled as an elastic, continuos beam. The beam model is rough but not wrong because the wing cell acts mainly as a beam. The rest of the wing doesn't carry heavy dynamic loads (Fig. 6). In regard to the real-time demands this wing model is an acceptable approximation.

The discrete wing model is obtained by introducing the first mode shapes of the flapping, hunting and torsional vibrations resulting from a previous Finite Element calculation of a most complex model. The use of mode shapes for discretizing a continuos system is called "Modal Reduction". It is very common in Structure Dynamics. The mode shapes include all information about the dynamic behavior of the structure. Modal reduced equation systems are much smaller than Finite Element equation systems, so their computation is much faster.

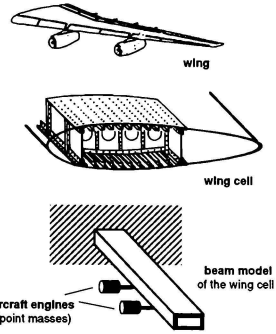


Fig. 6 Modeling of the wing

Introducing these mode shapes into a continuos differential equation leads to a system of differential equations that has only the unknown amplitudes of the considered mode shapes as DOF's. These amplitudes are not physical deflections. They are a sort of weighting factors. After each time-step in the real-time computation, the weighting factors are solved. The whole wing deflections can be calculated as a superposition of all considered modes.

4.1 The Principle of Virtual Work

The differential equations of the beam model are derived by using the principle of virtual work. The principle of virtual work is an energy formulation. It is also the basic for the Finite Element Method. In this energy formulation the real deflections are superposed by virtual deflections. These virtual deflections do work on the unknown real deflections and their mass, damp and stiffness forces. This leads to different virtual works, where virtual deformation work has to be distinguished from virtual load work.

The virtual deflections state is free selectable but it is clever to use the same mode shapes which are used for the real deflections. The virtual deflection work describes all the elasticity works. The virtual load work includes the inertia works and all the aerodynamic and gravitation works. Virtual deformation

work and virtual load work must arise an equivalence.

$$\delta V = \delta W$$

The principle considers continuos loads also as point loads. In the simple beam model both loads occur. There are continuos loads such as aerodynamic or gravitation loads and point loads such as discrete forces of the aircraft engines for example. In the same manner continuos stiffness' and inertial forces are considered together with point stiffness' and masses. The principle of virtual work can be written as:

$$\delta V = \int_0^b E(y)I(y)z_{el}''(y,t)\,\delta z''(y)dy$$

Here, the virtual deformation work of the flapping mode is written. It includes the Young's modulus and the geometrical moment of inertia. The virtual deformation work will be integrated over the wing span b. The product of these terms with the time-depending real deflections z_{el} gives a moment curve. The virtual deflections δz do work on this moment curve and produce the virtual deformation work.

The formulation for the virtual load work is given by:

$$\delta W = \int_0^b \mu(y,t)\ddot{z}_{el}(y,t)\delta z(y)dy + g\int_0^b \mu(y,t)\delta z(y)dy + \int_0^b p(y,t)\delta z(y)dy$$

For an easier understanding again only the terms of the flapping mode are shown. The virtual load work includes basically three terms. The first one describes the work of inertia, using the mass distribution μ and the real elastic accelerations of the wing $\ddot{z}_{el}$. The second term is usually static and considers the gravitation work using the gravitation constant g. In case of flight the fuel mass will be reduced. For that reason the mass distribution is a sum of structural and fuel mass. Both are time-dependent and so is the gravitation work. The third term includes all loads such as aerodynamic loads or engine thrust. These terms also depend on the local wing span coordinate and the time.

One can see another advances of the simple beam model. In the computation of the aerodynamic loads these forces are reduced to a single line (25%-line). If a complex 3-dimensional Finite Element model were used, the aerodynamic loads could not be reduced to a single line because this would lead to a singular line forces on the airfoil. In that case it would be necessary to smear the aerodynamic load line to a force area. That isn't necessary in the beam model.

Virtual deflections δz do work on the three terms above and produces a virtual load work.

4.2 Discretizing the Principle

Up to now the motion is described by continuos deflections and continuos system parameters such as the Young' modulus or the mass distribution for example. The whole model must be reduced to a small number of DOF's. The idea for discretizing the continuos model is an approximation of the real and virtual deflections by known deflection shapes. In this case the mode shapes of a complex Finite Element model are used. The different mode shapes are weighted by unknown factors and then superposed to the real deflection shape. The Finite Element model is calculated before each simulation, so there is no need to calculate it in every time-step. The number of the considered mode shapes depends not only on the accuracy that is wanted but also on the excited system frequencies in real flight situation. The mode shapes of higher eigenfrequencies don't influence the vibrations of the lower frequency level at normal flight conditions.

In the presented wing model the first mode shapes of the flapping , hunting and torsional vibrations are considered for descretizing the principle. The following equation represents the approximation of the real deflections by different mode shapes:

$$z_{el}(y,t) = \sum_{i=1}^{N} h_i(y)f_i(t) = \begin{pmatrix} h_1(y), \ldots, h_N(y) \end{pmatrix} \begin{pmatrix} f_1(t) \\ \vdots \\ f_N(t) \end{pmatrix} = \underline{h}^T \underline{f}$$

$h_i(y)$ describes the shape function of the mode shape number i with the wing span coordinate y. Its coefficient $f_i(t)$ is only time dependent. This is a so called weighting factor. It shows the strength of the influence of the mode shape number i on the whole vibration shape. Basically, this approximation leads to a separation of the time and the local coordinate dependencies.

Nearly the same approximation is used for the virtual deflections:

$$\delta z(y) = \sum_{k=1}^{N} h_k(y)\delta f_k = \begin{pmatrix} h_k(y), \ldots, h_N(y) \end{pmatrix} \begin{pmatrix} \delta f_k \\ \vdots \\ \delta f_N \end{pmatrix} = \underline{h}^T \underline{\delta f}$$

The difference lays in the type of the weighting factors. For the virtual deflections the weighting factors δf are virtual and don't depend on time. They remain free selectable.

After inserting the approximations above into the principle of virtual work, a big equation for the motion is given. This big equation is still an energy formulation but the time dependencies and the dependencies on local coordinates are separated now. In every integration term of this energy formulation there exist a virtual weighting factor δf_k. Sorting on these virtual factors leads to separate differential equations which have no virtual factors anymore.

$$\delta f_k : \sum_i \left[\int \mu h_k h_i dy \ddot{f}_i + \int EI h_k'' h_i'' dy f_i \right] - g \int \mu h_k dy - \int p h_k dy = 0$$

$$\sum_i \left[m_{ki} \ddot{f}_i + s_{ki} f - p_k \right] = 0$$

These differential equations can be written as a whole system of differential equations using matrix format after integration over the wing span coordinate y.

$$\underline{M}\,\ddot{\underline{f}} + \underline{S}\,\underline{f} = \underline{p}$$

$\underline{M}$: mass matrix
$\underline{S}$: stiffness matrix
$\underline{p}$: load vector.

It can be seen that no structural damping is used. In normal structures there isn't much structural damping. The main damping is brought by the aerodynamic forces considered by the load vector.

4.3 The System of Differential Equations of Motion

After derivation the energy formulation to the discrete model a coordinate system is needed for the description of motion. Fig. 7 shows the coordinates of a wing.

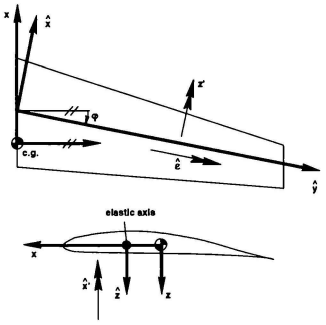


Fig. 7 Coordinate system of a wing

The elastic axis of a wing is singled out as a reference axis. A formulation of the equation relative to this reference axis will lead to a separation of flapping, hunting vibration and the torsional vibration. As a result the equations can be written much easier. Between a fuselage fixed coordinate system x, y, z and the wing fixed coordinate system $\hat{x},\hat{y},\hat{z}$ a constant angle φ (sweep) is sited. In addition the wing fixed coordinate system is displaced by the distance of gravity center and the wing center. The distortions of the flapping motion $\hat{z}''$ and the torsional motion $\hat{e}'$ which are used in the principle are also presented in Fig. 7.

The rest of the aircraft is modeled as a rigid body. The influence of the rigid body motion on the wing vibrations must be considered. For this reason transformation matrices for both wings are needed. The transformation matrix is given by:

$$\begin{pmatrix} \hat{x} \\ \hat{y} \\ \hat{z} \end{pmatrix} = \begin{bmatrix} \cos\varphi & \sin\varphi & 0 \\ -\sin\varphi & \cos\varphi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix}$$

Once again the virtual deformation work in the principle is presented, now including the deformation terms of the flapping, the hunting and the torsional motion. The rest of the principle cannot be shown. The equation is very large. Because the authors would like to show only the basic problems and steps to get the whole system of equations of motion they do not present every step in detail. One can read the full derivation in[5].

$$\delta V = \int EI_{\hat{x},FI}\, \hat{z}_{el}'' \delta z'' dy + \int GI_{\hat{t},FI}\, \hat{e}_{el}' \delta e' dy + \int EI_{\hat{z},FI}\, \hat{x}_{el}'' \delta x'' dy$$

The aerodynamic forces and the engine thrusts are directed to the aircraft length direction so they must be transformed into the rotated coordinate system of the wing.

One can find the excentricity of elastic axis to the gravitation axis of the wing in the inertial terms of the virtual load work.

The direction of gravitational force is always related to the actual position of the aircraft in motion. The Euler angles Φ and Θ take place in the principle.

It is very important to consider the relative kinematics of the system. All terms of second order time derivation in the principle must be completed by the relative motion between the aircraft and the earth.

At the end of all these derivations a system of differential equations of second order is given. This equation system has a small amount of DOF's, namely the weighting factors of the considered mode shapes. This system allows flightmechanical investigations in a full flight simulator that demands real-time computations.

The system of equations of second order including all important terms is given by:

$$\underline{M}\,\ddot{\underline{f}} + \dot{\underline{D}}\left(\underline{p}_D\right)\dot{\underline{f}} + \left[\underline{S} + \dot{\underline{S}}\left(\underline{p}_S\right)\right]\underline{f} = \underline{p}_0$$

For the real-time calculation, the system of second order will be transformed into a system of first order. The transformation doubles the amount of DOF's but this would be the same as one uses Finite Element Methods for modeling. One can see again: It is very important to reduce the amount of DOF's. Just the simple but necessary transformation to solve the equation system leads to a higher amount of DOF's.

$$\begin{pmatrix} \dot{\underline{f}} \\ \dot{\underline{f}} \end{pmatrix} = \begin{bmatrix} -\underline{\underline{M}}^{-1}\underline{\underline{D}} & -\underline{\underline{M}}^{-1}\left[\underline{\underline{S}}+\dot{\underline{\underline{S}}}\right] \\ \underline{\underline{I}} & 0 \end{bmatrix} \begin{pmatrix} \dot{\underline{f}} \\ \underline{f} \end{pmatrix} + \begin{pmatrix} -\underline{\underline{M}}^{-1}\underline{p}_0 \\ 0 \end{pmatrix}$$

The load vector $\underline{p}$ is separated to its aerodynamic parts.

$$\underline{p} = \underline{p}_0 + \underline{p}_S(\underline{f}) + \underline{p}_D(\dot{\underline{f}})$$

Fig. 8 shows the considered mode shapes for the beam model.

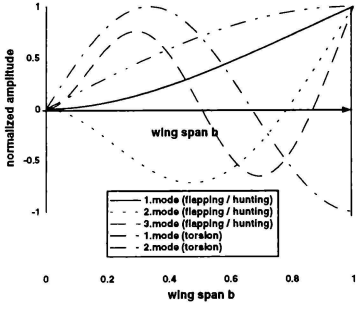


Fig. 8 Mode shapes for the beam model [5]

Conclusions

With exception of the instationary aerodynamics the above described models were integrated in the real-time simulation software of the A330/A340-FFS. The filter routines for the approximation of instationary effects had to be left because of the insufficient computation power of the simulation computer.

First results show significant aeroelastic effects, as had been expected. Fig. 9 shows the roll response to an aileron step input, both for a rigid and for a flexible wing. Dynamic flexible effects can only be stated in the roll rate. Roll acceleration and roll rate show higher frequency and reduced damping ratio of the Dutch roll motion for the flexible wing. The initial roll rate for the flexible wing is significantly lower than for the rigid wing, which means a distinctly reduced control effectiveness. This is clearly to be seen from the roll angle, which is reduced about 25% for the flexible wing. At the same time due to the reduced damping ratio the Dutch roll motion has larger amplitudes, which has a negative influence on passenger comfort. It has to be mentioned that no absolute values should be taken from the shown graphs because the data basis for the structure flexible wing is preliminary. Nevertheless the results give the correct quantitative behaviour.

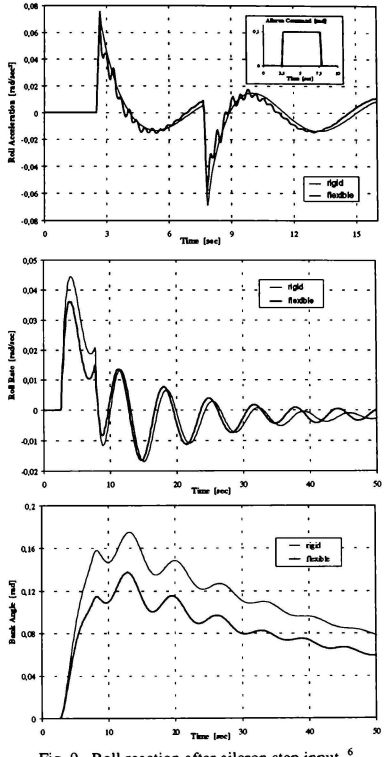


Fig. 9 Roll reaction after aileron step input [6]

These preliminary results show that the main effects of structural elasticity on aircraft motion are well represented by the implemented model:

(1) reduction of control effectiveness and

(2) coupling effects with the flightmechanical modes.

The development of the real-time model of the flexible aircraft is to be continued. The actual work is related to extend the degree of considered flexible modes of the wing and to take into account flexibility of fuselage and tail plane unit.

A new simulation computer is to be integrated in the A340-FFS to meet the computation requirements for the implementation of instationary effects. Furthermore the aerodynamic model will be extended to reproduce high lift configurations. This simulation model will be the basis for further research work on the development of new control laws and flying quality studies.

References

[1] O. Schmidt — Entwicklung eines nichtlinearen Flugzeug-Simulationsprogramms und Integration in die Simulationsumgebung SIMEX des A340-Simulators des Zentrums für Flugsimulation Berlin
Study at the Aerospace Institute, TU-Berlin, 1996

[2] M. Dahms — Erstellung eines Programmpaketes zur Bestimmung der aerodynamischen Beiwerte eines generischen Flugzeuges für die Echtzeit-Flugsimulation
Study at the Aerospace Institute, TU-Berlin, 1996

[3] J. Buxbaum — Modellierung von Querrudereinflüssen für die Echtzeit- Flugsimulation eines elastischen Flugzeugs
Thesis at the Aerospace Institute, TU-Berlin, 1997

[4] G. Zrenner — Approximation instationärer Luftkräfte eines Tragflügels für die Echtzeit-Flugsimulation
Thesis at the Aerospace Institute, TU-Berlin, 1996

[5] G. Duus — Modellierung des aeroelastischen Verhaltens eines Tragflügels für die Echtzeit-Flugsimulation
Thesis at the Aerospace Institute, TU-Berlin, 1996

[6] O. Schmidt — Untersuchungen zum Einfluß dynamischer Flügelverformungen auf das flugmechanische Verhalten eines sehr großen Transportflugzeuges
Thesis at the Aerospace Institute, TU-Berlin, 1996

[7] W. Kindel, R. Liebich — Modellierung eines elastischen Flügels für die Echtzeit-Flugsimulation
DGLR annual meeting, Dresden, 1996

MODELLING OF AIRCRAFT FLIGHT BY MEANS OF DYNAMICALLY SIMILAR MODELS WITH A FLIGHT CONTROL SYSTEMS SIMILARITY

S. Sadovnitchii, Ph.D., Instituto Politecnico Nacional ESIME-Ticoman, Mexico D.F., Mexico
V. Ryabkov, Dr. Tech. Science, Kharkov Aviation Institute, Kharkov, Ukraine.
A. Rushenko, Dr. Tech. Science, Kharkov Aviation Institute, Kharkov, Ukraine.
J. Sandoval, Instituto Politecnico Nacional ESIME-Ticoman, Mexico D.F., Mexico

ABSTRACT

This paper deals with the methods of physical modelling of Flight Control Systems (FCS) by means of Dynamically Similar free-flying Models (DSM) for the investigation of stability and controllability of aircraft at subsonic flight speeds. The subsonic flight regime allows us to avoid Mach number similarity considerations. The large scale of the DSM meets auto similarity of Reynolds numbers, whilst Froude similarity is assured during the development and manufacturing the DSM. This paper proves the presence of necessary and sufficient conditions of similarity for the FCS of an aircraft, and that of its DSM. The generalized scale coefficients for transitioning from the FCS's aircraft gain factors, to those of the FCS of the DSM have been obtained, and it is shown that the coefficients depend only on the linear scale of the DSM.

Keywords:

modelling, aircraft, scale, coefficients, autosimilarity.

The modelling of aircraft dynamics is one important problem when designing a new Flight Control Systems (FCS). When applied in the early stages of design, mathematical simulation provides considerable advantages, as well as some drawbacks; among the latter is the influence that the assumptions introduced in the preparation of the aircraft's mathematical model has on the final results. That is, theoretical methods always require an experimental validation of the basic assumptions.

Wind-tunnel test and other traditional methods of experimental aerodynamics does not adequately provide the reproduction of complex aircraft dynamics. Aircraft flight testing is expensive and presents a risk for test-pilots. Furthermore, test can only provide results after the production of the first prototype aircraft.

Large scale dynamically similar free-flying models have been widely used during the creation of new flying vehicles, for testing original aerodynamic concepts and exploring high risk flight envelopes.

They are also utilized for flight dynamics investigations, for the determination of their optimum control laws, as well as for aeroelasticity investigations[1,2]

The main advantages of this method are:

a) The unlimited space that surrounds the model (it excludes the interference between the walls and boundary layer of the flow inside a wind tunnel);
b) The absence of a suspension and attachments which distorts natural oscillations of the model and introduce additional masses and friction;
c) Last but not least, the possibility of investigating the most dangerous regimes of flight, before the first prototype of the aircraft under development is created, as well as saving large investments and time for the construction of unique installations (such as special wind tunnels).

The methodological possibilities of experimenting by means of DSM, represents a highly effective scientific and research tool.

A DSM is an autonomous, pilotless, research and development multiple mission aircraft. It can carry out automatic o remotely controlled flights, according to a pre-set program, and has the capability to record all flight relevant information during a test. The DSM is geometrically similar to the aircraft under investigation. Its mass, moment of inertia and other parameters ensure exact performance similarity, including the case when we need to ensure an adequate simulation of aeroelastic processes. DSM is a physical model of an aircraft, and it excludes all errors that the mathematical model of that aircraft may have. It is of most importance in the investigation of new flight regimes: for example manoeuvre such as the well known "Pugachev Cobra" (reaching an angle of attack of more than 90° without stalling). The

"Copyringht © 1997 by the
Ameruican Institute of Aeronautics and
Astronautics,Inc. All rights reserved."

development of a mathematical model for predicting aircraft's behaviour is as mentioned above a very difficult problem, however, a DSM may solve this problem readily. Regimes which attain an of angle of attack equal to 90° are pre-set at the FCS of the DSM, and after flight tests, better control laws may be selected. The DSM is launched from a carrier (aircraft or helicopter) or from a ground ramps with the help of a solid-propellant ascent stage.

After stage separation, the DSM accomplishes pre-programmed manoeuvres; in case of an emergency, or after flight program completion, a parachute and soft landing system performs a proper recovery. It may also recover the DSM from critical or uncontrolled flight regimes, can stabilize and decelerate its motion and at the landing moment, the same system will decrease vertical speed for touch-down and counteracts ground wind effects. After processing the flight's information, performing turn-round servicing and on-board systems readjusting, in accordance with a new flight program, further flight experiments may be conducted. Thus, dictated by issues of test pilot safety as well as by economic considerations, there is a clear need for the most realistic simulation of the dynamics of flight, specially in the exploration of critical flight regimes for new aircraft, by means of such type of dynamically similar free-flying models.

The determination of main dimensional flight parameters though DSM procedures results, usually beyond doubt, however, as a rule, only non-dimensional parameters require special investigations. As denoted in the literature, the non-dimensional parameters of process shows invariance under metric transformations and may be utilized as an independent similarity criterion. But such simplified formal approach to satisfy similarity criteria in non-dimensional parameters is only valid within relatively simple problems of similarity and can not be used for tests on DSM.

An example of non-dimensional parameters in which simulation modelling requires special analysis is the angle of attack α (if it is measured in radians). There are no doubts about the need of an adequate representation of this angle, but only after conducting special investigations it becomes clear what values for mass, autopilot coefficients and other should be taken into account during DSM's design. Under a simulation of steady-state flight conditions in wind tunnels the requirement on the angle of attack equivalence can be achieved relatively easily: it is enough to fix the model in the required position with respect to the flow. Moreover, force unbalances, acting on the model may be compensated by actuators at the supports.

In tests of DSM the angle of attack is determined by the autopilot's sensors by means of aerodynamic, inertial and gravity forces which act on the model. Therefore, the task of providing an equivalence between the model's angle of attack α_M, and the real aircraft's α_N, at first, can be related to an adequate representation of the model's parameters that are contained in the criterion of dynamic similarity of Newton laws and Froude and Mach numbers. Let us now analyse this relationship.

It is well known that it is impossible to provide for full aerodynamic similarity in a single test's flight, in other words, simultaneous equality of Reynolds, Froude and Mach numbers for the real aircraft, and for the DSM is not within reach. However, we can neglect similarity of Mach number for the simulation of stability and controllability, when the airspeed is lower than the velocity of sound. In the case of a large scale DSM model we can demonstrate that we have self similarity of Reynolds number. However, for the exploration of stability and controllability we must observe first of all the Froude criterion. The equation expressing the relationship between inertial and weight forces, follows:

$$\frac{V_N^2}{gL_N} = \frac{V_M^2}{gL_M} = Fr; \qquad Fr = idem$$

where V is the velocity, g the acceleration due to gravity, and L is the length.

Under DSM experiments the task of providing kinematic similarity assumes two approaches:

1.- It is necessary for similarity conditions to equalize the magnitude of the angle of attack, that determines the forces acting on the model, and the corresponding model trajectory.
2.- It is necessary for satisfying similarity that the model's trajectory, for example straight horizontal flight (SHF), and forces acting on the model, and model's angle of attack correspond to this trajectory.

Let us examine both approaches in experiments in the case when the Froude criterion is satisfied. Together with above mentioned we should note that for a model that shows accordance with Newton's criterion:

$$Ne = \frac{m}{\rho \cdot L^3},$$

mass $m_M = K_\rho \cdot K_L^3 \cdot m_N$;

acts the of weight forces

$$W_M = m_M \cdot g = K_\rho \cdot K_L^3 \cdot m_N \cdot g = \\ = K_\rho \cdot K_L^3 \cdot W_N \qquad (1)$$

where $K_L = \frac{L_M}{L_N}; K_\rho = \frac{\rho_M}{\rho_N}$ are the scales of linear dimensions and represents the density of the environment.

As soon as $Z_N = C_{LN} \cdot \frac{\rho_N \cdot V_N^2}{2} \cdot S_N$

$Z_M = C_{LM} \cdot \frac{\rho_M \cdot V_M^2}{2} \cdot S_M$; the lift force of the model in case of equivalence of angles of attack α and lift coefficients is

$$Z_M = K_\rho \cdot K_V^2 \cdot K_L^2 \cdot Z_N \qquad (2)$$

where $K_V = \frac{V_M}{V_N}$ is the scale of the velocity, S is the bearing surface.

The simple flight regime being modeled (SHF), that is in the absence of the vertical component of thrust, is characterized by the condition

$$Z_N = W_N . \qquad (3)$$

For obvious reasons $g_M = g_N$ and by satisfying the Froude criterion $(K_V = \sqrt{K_L})$ substitute in formula (3) expression (1) and (2) making equation $Z_M = W_M$, i. e. SHF of aircraft with Fr=idem is represented by the same model's angle of attack is equal to the aircraft. For maneuvering, aircraft's modelling in a certain point of the trajectory we find the overload on the model table:

$$n_{zM} = \frac{Z_M}{W_M} = \frac{K_\rho \cdot K_V^2 \cdot K_L^2 \cdot Z_N}{K_\rho \cdot K_L^3 \cdot W_N}\Big|_{K_V = \sqrt{K_L}} = \\ = n_{zN}; \qquad (4)$$

and the curvature of model's trajectory is

$$R_M = \frac{m_M \cdot V_M^2}{Z_M - W_M} = \frac{V_M^2}{(n_z - 1) \cdot g} = \frac{m_V \cdot K_V^2 V_N^2}{Z_N - W_N} = \\ = K_V^2 \cdot R_N\Big|_{K_V = \sqrt{K_L}} = K_L \cdot R_N . \qquad (5)$$

Hence, with Froude and Newton criterion satisfied, both of the above methods are equivalent: the action of aerodynamics forces on the DSM causes the geometry of the trajectory to be the same, in the model and in the real aircraft, when their angles of attack are equal.

It is also possible to show that, without satisfying the Froude criterion (for example satisfying the Mach criterion ($M = \frac{V}{a}$), and having $\alpha_M = \alpha_N$, it is necessary to move along a curvilinear trajectory with an overload on the model given by:

$$n_{zM} = \frac{K_V^2}{K_L} \cdot n_{zN} . \qquad (6)$$

That is, satisfying M=idem $(K_V \approx 1)$ and the scale $K_L = \frac{1}{5}$, the simulation of violent maneuvers with $n_{zN} = 8$, would require a normal load factor $n_{zM} = 40$; and that with a safety coefficient of f=1.5 will force on the designer a normal load factor of $n_{zM} = 60$. To design such model is in practice an unrealizable feat.

However, theoretical, methodological and designer efforts allow us to produce a DSM that provides an adequate reproduction of complex aircraft dynamics. Fig. 1 shows the result of spin flight test of a Sukhoi SU-27 aircraft and of its DSM. This diagram shows a close convergence of flight test results. Furthermore, necessary and sufficient conditions of similarity must also be meet for the FCS of an aircraft, and that of its DSM.

At an early stage in the development of free-flying models, investigators were only interested in the aerodynamic characteristics of the aircraft, without taking much into account the influence of the FCS on these characteristics. Thus, modelling of a FCS through DSM were not conducted, as far as we know [1,2]. Nevertheless the

frequent utilization as well as the growth of research efforts on active control systems has provided a need to further simulate FCS; specially since a similarity transfer function between the FCS of the real aircraft, with that of the DSM, is still incomplete or simply absent.

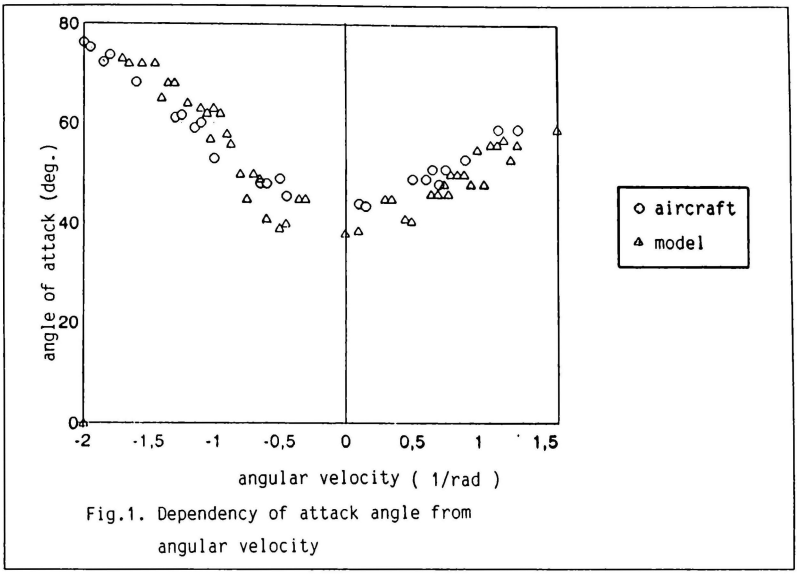


Fig.1. Dependency of attack angle from angular velocity

Necessary conditions also includes the presence of a similarity criterion which must be numerically equivalent for similar phenomenon (π-Theorem from the theory of dimensions)[3] sufficient conditions in geometrical similarity of the aircraft and its model, and similarity of the initial and boundary conditions of the processes to be compared.

To ensure sufficiency of similarity conditions, at first, the DSM must be manufactured as an exact copy of the real aircraft, but on a smaller scale, this provides the first condition of sufficiency, that is, similarity of geometrical parameters between the model and the real object. Second, to satisfy the functional similarity of the block diagram of the FCS, i.e. to keep in the DSM formulation the same control laws as on the real aircraft; this is usually not a problem. In the case of the FCS of the designed DSM, control laws are not selected, but are taken directly from the real aircraft. Third, ensuring similarity of the starting boundary conditions of confronted phenomenon: the flight of the real aircraft and its model occur both in the real atmosphere.

To satisfy all necessary similarity conditions is a more complex question. For this it is necessary to obtain scale transition coefficients from the aircraft's FCS gain factors, to those from the FCS of the DSM. These coefficients can be derived by means of a formal mathematical method in which we apply the rules of the theory of dimensions. In this case it is necessary to examine the differential equations of aircraft displacement.

The method of reception of necessary conditions for similarity is valid for n^{th} order equations. For a clear demonstration of necessary conditions for similarity between of the FCS of an aircraft and that of its DSM we may examine the

longitudinal, short-period motion of the system "Aircraft-FCS", described by the following equations[6]:

$$(s^2 + C_1 s)\theta + (C_5 s + C_2)\alpha + C_3\delta = 0$$
$$-s\theta + (s + C_4)\alpha = 0 \qquad (7)$$
$$\delta = G_\theta \cdot \theta + G_q \theta s + \frac{1}{s}\theta G_{\bar{\theta}}$$

where s is the variable of the Laplace transformation, θ the pitch angle, δ is the surface deflection, G the gain of FCS. C1...C5 are coefficients of linearized equations .

Let us write the system of equations (7) in the form

$$\varphi_1 + \varphi_2 + \varphi_3 + \ldots + \varphi_{12} = 0 \qquad (8)$$

Where

$$\varphi_1 = s^4\theta, \qquad \varphi_2 = s^3 C_4\theta,$$
$$\varphi_3 = s^3 C_1\theta, \qquad \varphi_4 = s^2 C_1 C_4\theta,$$
$$\varphi_5 = s^3 C_5\theta, \qquad \varphi_6 = s^2 C_2\theta, \quad \varphi_7 = s^3 C_3 G_q\theta,$$
$$\varphi_8 = s^2 C_3 C_4 G_q\theta, \qquad (9)$$
$$\varphi_9 = s^2 C_3 G_\theta\theta; \quad \varphi_{10} = sC_3 C_4 G_\theta\theta,$$
$$\varphi_{11} = sC_3 G_{\bar{\theta}}\theta, \quad \varphi_{12} = C_3 C_4 G_{\bar{\theta}}\theta.$$

If the terms of the equation contain symbol of differentiation and integration, by consideration of proportionality condition it is possible to omit these symbols (as they have not dimension and do not influence of proportionality condition)[7] Hence, we can replace the appropriate terms of equations by their analogues φ', which is possible to name "integral-analog", i.e. $d^n x / d^n y$ to replace on x / y^n and $\int xdy$ to replace on xy.

In order to comply with the similarity criterion by the integral-analog method, we write:

$$\varphi_1' = \frac{\theta}{t^4}; \qquad \varphi_2' = \frac{C_4\theta}{t^3}; \quad \varphi_3' = \frac{C_1\theta}{t^3};$$
$$\varphi_4' = \frac{C_1 C_4\theta}{t^2} \qquad \ldots$$
$$\varphi_9' = \frac{C_3 G_\theta\theta}{t^2}; \quad \varphi_{10}' = \frac{C_3 C_4 G_\theta\theta}{t};$$
$$\varphi_{11}' = \frac{C_3 G_{\bar{\theta}}\theta}{t}; \quad \varphi_{12}' = C_3 C_4 G_{\bar{\theta}}\theta \qquad (10)$$

and

$$\pi_1 = \frac{\varphi_2'}{\varphi_1'} = C_4 t; \qquad \pi_2 = \frac{\varphi_3'}{\varphi_1'} = C_1 t;$$
$$\pi_3 = \frac{\varphi_4'}{\varphi_1'} = C_1 C_4 t^2;$$

$$\pi_9 = \frac{\varphi_{10}'}{\varphi_1'} = C_3 C_4 G_\theta t^3; \quad \pi_{10} = \frac{\varphi_{11}'}{\varphi_1'} = C_3 G_{\bar{\theta}} t^3;$$
$$\pi_{11} = \frac{\varphi_{12}'}{\varphi_1'} = C_3 C_4 G_{\bar{\theta}} t^4 \qquad (11)$$

As far as the similarity criterion these are non-dimensional quantities, therefore the product of the scale coefficients of this criterion must be equal unity:

$$K_{C4} \cdot K_t = 1; \qquad K_{C1} \cdot K_t = 1;$$
$$K_{C1} \cdot K_{C4} \cdot K_t^2 = 1; \qquad K_{C5} \cdot K_t = 1;$$
$$K_{C2} \cdot K_t^2 = 1;$$

$$K_{C3} \cdot K_{C4} \cdot K_{G\theta} \cdot K_t^3 = 1; \quad K_{C3} \cdot K_{G\bar{\theta}} \cdot K_t^3 = 1;$$
$$K_{C3} \cdot K_{C4} \cdot K_{G\bar{\theta}} \cdot K_t^4 = 1. \qquad (12)$$

With equation (12) and $K_t = \sqrt{K_L}$, we may calculate the scale coefficients for equation (7)

$$K_{C1} = K_{C4} = K_{C5} = 1/\sqrt{K_L};$$
$$K_{C2} = K_{C3} = 1/K_L;$$
$$K_{G\theta} = 1; \qquad (13)$$
$$K_{Gq} = \sqrt{K_L};$$
$$K_{G\bar{\theta}} = 1/\sqrt{K_L} \text{ .a1}$$

The foregoing method of obtaining the scale coefficients is a formal mathematical method, therefore it is necessary to verify by means of other procedures, which do take into account the physical nature of the occurring process[4].

The following equations are valid, if similarity of flows is complied[5]:

Length	$L_M = L_N \cdot K_L$
Area	$S_M = S_N \cdot K_L^2$
Mass	$m_M = m_N \cdot K_L^3$

Density $\rho_M = \rho_N \cdot K_\rho$ (14)

Acceleration $a_M = a_N,\ K_a = 1$

Angle $\varphi_M = \varphi_N,\ K_\varphi = 1$

Angular velocity $\omega_M = \omega_N / \sqrt{K_L}$

Moment of inertia $I_M = I_N \cdot K_L^5$

Observance of Froude criterion completes these equations:

Time $t_M = t_N \cdot K_t = t_N \cdot \sqrt{K_L}$

Velocity

$$V_M = V_N \cdot K_V = V_N \cdot \sqrt{K_L} \tag{15}$$

It is know that the expression for obtaining the point of pitch damper[6]

$$G_q = \frac{2\xi\sqrt{C_2 + C_1 C_4} - (C_1 + C_4 + C_5)}{C_3} \tag{16}$$

where ξ is the damping ratio.

The coefficients C1...C5 are defined by the following equations [6]:

$$C_1 = -\frac{C_{M\bar{q}}}{I_Y}\rho\frac{V}{2}S\bar{c}^2\left[\frac{1}{s}\right];$$

$$C_2 = -\frac{C_{M\alpha}}{I_Y}\rho\frac{V^2}{2}S\bar{c}\left[\frac{1}{s^2}\right];$$

$$C_3 = -\frac{C_{M\delta}}{I_Y}\rho\frac{V^2}{2}S\bar{c}^2\left[\frac{1}{s^2}\right];$$

$$C_4 = -\frac{C_{L\alpha} + C_D}{m}\rho\frac{V}{2}S\left[\frac{1}{s}\right]; \tag{17}$$

$$C_5 = -\frac{C_{M\dot{\alpha}}}{I_Y}\rho\frac{V}{2}S\bar{c}^2\left[\frac{1}{s}\right],$$

where C_M is the pitching moment, $C_{M\bar{q}} = \partial C_M / \partial\bar{q}$, $C_{M\alpha} = \partial C_M / \partial\alpha$, $C_{M\delta} = \partial C_M / \partial\delta$, $C_{M\dot{\alpha}} = \partial C_M / \partial\dot{\alpha}$, $\bar{c}$ is the mean aerodynamic chord, I_Y is moment of inertia about the Y body axis, [ ] designation of dimensions of a quantity.

With equation (14 and 15) solved, the coefficients (17) of linearized equations for the system "model-damper" will be:

$$C_{1M} = \frac{C_{M\bar{q}}}{I_{YN}K_L^5}\ \frac{\rho_N V_N\sqrt{K_L}}{2}\ S_N K_L^2 \bar{c}_N^{-1} K_t^2 =$$

$$= \frac{1}{\sqrt{K_L}} C_{1N} \tag{18}$$

Analogously, we may obtain:

$$C_{2M} = \frac{1}{K_L}C_{2N};\quad C_{3M} = \frac{1}{K_L}C_{3N};$$

$$C_{4M} = \frac{1}{\sqrt{K_L}}C_{4N};\quad C_{5M} = \frac{1}{\sqrt{K_L}}C_{5N}; \tag{19}$$

Therefore, with this equation solved, the gain of the pitch damper for the DSM will be:

$$G_{qM} = \frac{2\xi\sqrt{\frac{1}{K_L}C_{2N} + \frac{1}{\sqrt{K_L}}C_{1N}\frac{1}{\sqrt{K_L}}C_{4N}}}{\frac{1}{K_L}C_{3N}} -$$

$$- \frac{\frac{1}{\sqrt{K_L}}C_{1N} - \frac{1}{\sqrt{K_L}}C_{4N} - \frac{1}{\sqrt{K_L}}C_{5N}}{\frac{1}{K_L}C_{3N}};$$

$$G_{qM} = \sqrt{K_L}G_{qN} \tag{20}$$

In a similar way, the transformation for roll and yaw dampers, have the same relation between the gain of the aircraft and that of the DSM.

If we take the control law for longitude zero-constant-error, as follows:

$$\delta = G_q \cdot \dot{\theta} + G_\theta \cdot \theta + G_{\tilde{\theta}} \cdot \int \theta \cdot dt \tag{21}$$

where

$$G_\theta\left[\frac{\text{degree of elevator}}{\text{degree of pitch angle}}\right]$$

$$G_{\bar{\theta}}\left[\frac{\text{degrees of elevator / s}}{\text{degree of pitch angle}}\right]$$

and make an analogous transformation, we then obtain,

$$G_{qM} = \sqrt{K_L}\,G_{qN}; \qquad G_{\theta M} = G_{\theta N};$$

$$G_{\bar{\theta}M} = \frac{1}{\sqrt{K_L}}G_{\bar{\theta}N} \tag{22}$$

Therefore, we can consider that the presence of necessary conditions of similarity for the FCS of the aircraft and that of the DSM has been proven.

It is self evident, that the value and dimensions of the gains are determined by means of coefficients from the equations for the system "Aircraft-FCS". Since these coefficients for the DSM and for the aircraft have a certain relation, all transformations in the equations, from aircraft to model's gains, remain only a matter of scale, which defines the dimensions of its gains. Thus, in all cases, the coefficients to change from aircraft to model's gains, depend on actual dimensions and are defined by the following equations:

$$K = 1, \text{ for dimension } \left[\frac{\text{degree}}{\text{degree}}\right]$$

$$K = \sqrt{K_L}, \text{ for dimension } \left[\frac{\text{degree}}{\text{degree / s}}\right]$$

$$K = 1/\sqrt{K_L}, \text{ for dimension } \left[\frac{\text{degree / s}}{\text{degree}}\right]$$

Conclusions

We believe physical modelling of a FCS by means of large scale dynamically similar free-flying models has a long-range perspective, and allow far significant increase in the quality of design of new types of aircraft, and its FCS; with the advantage of drawing nearer the physical model to the natural object simulated.

The results show the possibility of physically modelling the dynamics of flight, taking into account all peculiarities of aerodynamics, block diagram equivalence of FCS, and real atmospheric conditions.

References:

1.- Patuxen River.Md, "NASA Drop Model Sinks", Aviation Week & Space Technology, January 20, 1997. P. 59

2.- Petersen, K. L. "Flight control system development of HiMAT vehicle" AIAA, Paper N81-17989. pp 196-197

3.- Durand, W. F. "Aerodynamic Theory". Vol. 1, Peter Smith, 1976, pp 23-30

4.- Tcheranovsky, Q. R., Sadovnitchii, S. N., Lavrinenko, G. V. "Some questions related to modelling of aircraft by means of utilizing large scale free-flying dynamically similar models". Journal of Academy of Sciences of USSR "Voprosy Kibernetiki" Serie "Simulation of Control processing man-machine systems", 1988, pp 110-119 (in Russian)

5.- Durand, W. F. "Aerodynamic Theory" Vol. VI, Peter Smith, 1976, pp 203-204.

6.- Mikhalyov, I. A. Okoyemov, B. N., Tchikylaev, M. C. "Aircraft Automatic Control Systems". Machinostroyonie, Moscow, 1987 pp. 51, 129 (in Russian)

7.- Vyenikov, V. A., Vyenicov, G. V., "The Theory of Similarity and Modelling". Machinostroyenie, Moscow. 1984. (in russian).

AIAA-97-3793

CAD Embedded CAE Tools for Aircraft Designers as Applied to Landing Gear

Matthew S. Schmidt,* Chris Paulson†

CADSI, Mid-Atlantic Region
One Fountain Square
11911 Freedom Dr., Suite 590
Reston, VA 20190

CADSI, Corporate
2651 Crosspark Road
Coralville, IA 52241

ABSTRACT

In the aerospace community the perception of quality no longer relates to only whether the product performs to specification. The perception now includes whether or not the hardware was produced "on-time and on-cost". To live up to the expanding expectations and maintain a competitive stance, design cycles times have to be reduced to maintain sound control on program costs and schedules. As the continuous change occurs, traditional analytical approaches are being scrutinized for streamlined efficiency. When shown to be antiquated, new approaches are adopted to rise to the challenge. A major challenge has been to transform old paradigms into new paradigms. Once such change has been to remove the barriers between disciplines and form product development teams to encourage cross-pollination of engineering activities. As part of the natural evolution of the teams, CAE tools have to become embedded within the CAD environment. This embedding of tools tremendously reduces duplication of effort, redundant data bases, amount of coordination, and thereby program costs.

The foundation of the paper being presented here is the application of CAD embedded CAE tools. The "new paradigm" that will be demonstrated by this paper is the Simulation Driven Design (SDD) environment. The application example that will be used is a landing gear exposed to the following environments: variable speed drop test and retraction. The CAD tool that will be applied is CATIA. CATIA provides a solid modeling environment for designing components and mechanical assemblies. The CAE tools that will be applied are CATDADS, PolyFEM and EASY5. CATDADS is a tool that is used to predict the behavior of mechanical assemblies. The equations of motion are automatically formed and solved. Positions, velocities, accelerations and reaction loads are predicted for all components in the assembly. PolyFEM is a finite element analysis package that includes interfaces to CAD products, an automatic mesher, and a p-type finite element solver. EASY5 is a control design tool which distinguishes itself from other tools on the market by its hydraulics libraries. The CAD/CAE tools are used in conjunction to achieve the desired "total system prototype" prior to any physical devices being constructed.

1.0 Introduction

The foundation of the paper being presented here is the application of CAD embedded CAE tools. The "new paradigm" that will be demonstrated by this paper is the Simulation Driven Design (SDD) environment. SDD is a tool applied by designers, analysts and system engineers alike. The SDD is the manner in which the "total system prototype" is achieved electronically within the CAD environment prior to the development of any physical apparatus. The "total system prototype" operates on the CAD systems solids and assembly database. The electronic prototypes considered by the SDD are assembly, control/assembly and component. Integrating tools with the CAD programs have been a move in the right direction to improve

*CADSI Senior Applications Engineer, 703-925-9743
†CADSI Applications Engineer, 319-626-6700

Copyright © 1997 by the American Institute of Aeronautics and Astronautics, Inc.

efficiency and reduce cost; however, to a large extent databases are still being duplicated, too much training is required to use the integrated products, and too much non-value added activity is needed to assist in each phase of the integration. SDD is the embedding of the motion and structural analysis tools into the design environment such that CAD/CAE activities occur in parallel with the same database. By embedding CAE tools the results of the engineering process are naturally available for upstream processing in the CAD environment. Simply stated, SDD allows the engineer to electronically prototype during the design process ensuring that the final product is the most efficient compromise possible. Physical testing then becomes a tool for final verification rather than a part of the iterative design process.

The CAD tool that will be applied is CATIA. CATIA provides a solid modeling environment for designing components and mechanical assemblies. The CAE tools that will be applied are CATDADS, PolyFEM and CATDADS/Plant EASY5. As part of the SDD, the "look and feel" of the CAD package is maintained by the menu items and functionality of the CAE embedded tools. CATDADS operates directly off of the CAD systems solids and assembly database. The software is a tool that is used to predict the behavior of a mechanical assembly.[1] Force elements such as actuators, tires and controls can be added to the assemblies within this tool. The equations of motion are automatically formed and solved. Positions, velocities, accelerations and reaction loads are predicted for all components in the assembly. The results are available for visual animation and graphical plotting. PolyFEM is a finite element analysis package that includes interfaces to CAD products, an automatic mesher, and a p-type finite element solver. [2] The software is design engineer oriented and geometry based. EASY5 is a control design tool which distinguishes itself from other tools on the market by its hydraulics libraries. [3] The control models are assembled from a variety of components in hydraulics, mechanical, multi-phase fluid, pneumatic and thermal libraries.

The application example that will be presented in this SDD paper is a landing gear exposed to the following environments: variable speed drop test and retraction. The paper will demonstrate the application of CATDADS, PolyFEM and CATDADS/Plant EASY5 (CAE tools) while embedded within CATIA (CAD tool). The CAE tools will be used to understand the performance and requirements of the landing gear without duplicating the CAD database. Each phase required to go from the CAD tool to each of the CAE tools as applied to the landing gear will be demonstrated by this paper without ever having to leave the CAD environment. Above all else, this is critical because any changes made to the system are immediately available for upstream processing.

2.0 Total System Prototype

Figure 1 - Flowchart and Process for Applying SDD

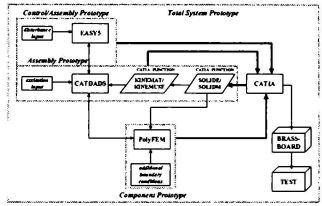


Figure 1 shows the flowchart and process for the application of SDD. CATIA is shown to be the beginning and the end of the SDD environment. The entire flowchart represents a model for the "total system prototype". The overall prototype consists of an assembly, control/assembly, and component prototype. Each of the prototypes will be presented in detail in subsequent sections. The regions enclosed by dashed lines indicate sub-categories of the "total system prototype". The single lines connecting the boxes show the flow of information from the CATIA database. The double lines show the upstream processing path for design modifications back to the design database from each of the prototyping categories. The core CATIA functions needed for this prototyping environment are SOLIDE, SOLIDM, KINEMAT and KINEMUSE. These functions will be described in the next section.

The strength of SDD is having the CAE tools embedded inside of the CAD tools such that a true "total system prototype" can be achieved by allowing product development teams (PDT) take full advantage of the design database. SDD facilitates the mission of PDTs by merging the system, design and analysis databases. As the

design modifications are made, and due to the manner in which the tools are embedded, the prototyping automatically reflects the changes. Therefore, design trade studies, performance assessments and complex system behavior can be quickly and efficiently investigated by the PDT without having to depart from the CAD system.

3.0 CATIA

The prototyping components are designed using CATIA exact and mock-up solids (SOLIDE/SOLIDM). The exact solids provide the precise geometric representation of each part, while mock-up solids provide a faceted geometric representation. The designed parts are arranged into different sets. From a mass and inertia point of view, all of the parts within a CATIA set are rigidified such that the composite characteristics are available to the assembly prototyping. The KINEMAT function allows the different geometric sets to be connected by appropriate joint types. As the boundary conditions are specified, ICONS (indicate condition) appear on the assembly indicating the joint location and type. The joints allow articulation or relative motion to occur between the geometric sets. The KINEMUSE function allows prescribed motion to be super-imposed onto the sets which are kinematically coupled. The bi-product of the solution is the system motion in which part interference's and collisions can be checked. As a side note, the output of a kinematic solution is purely related to the position, velocity and acceleration of the system components which results from prescribed motion. It is believed that the reader understands the fact that reaction loads are not a bi-product of kinematic analysis.

Two types of kinematic sets can be defined in KINEMAT: complete and incomplete systems. A complete system is one in which all the systems paths of least resistance (degrees of freedom) are controlled by joint boundary conditions and prescribed motion. In other words, a complete system has zero degrees of freedom. Only complete systems can be solved with the CATIA KINEMUSE function. To solve an incomplete system, one in which degrees of freedom exist, the analysis type has to switch from kinematic to one that also includes dynamics. The CATDADS function provides this additional capability. CATDADS is the trade name for the DADS software embedded in the CATIA environment. In order to use CATDADS, the CATIA installation must first have authorization for the KINEMAT function. CATDADS requires additional information beyond the system kinematics, which is readily available in the solids database: mass, center of mass location, and inertia tensor. Furthermore, the CATDADS entities are stored within the CATIA model and can be modified.

4.0 CATDADS

Figure 2 - Assembly Prototype Process

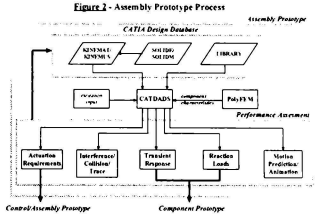


Figure 2 shows an expanded view of the "assembly prototype" referenced by Figure 1. The "assembly prototyping" occurs both in KINEMUSE and CATDADS. CATDADS provides the kinematic/dynamic response of the system as a result of load transmission through the systems kinematic chain. The system can contain various types of force elements, externally applied loads, motion drivers, etc., and the resulting descriptive equations of motion are integrated forward in time. The system equations of motion are automatically formed and solved. Positions, velocities, accelerations and reaction loads are predicted for all components in the assembly. The CATDADS analysis executables can be customized by the engineers to add additional capability to the commercial software as needed. The solution types available within CATDADS are kinematic, inverse dynamic, dynamic and static. As has been previously stated, reaction loads are recoverable in all the analysis types except for kinematics.

The starting point for the "assembly analysis" is from CATIA KINEMAT/ KINEMUSE functions in which the sets are kinematically coupled and motion drivers applied to the system such that the system is complete. Once the KINEMAT set, complete or incomplete, has been represented, CATDADS is invoked and the kinematic mechanism is

automatically converted to a CATDADS model. Multiple kinematic mechanisms can be managed by the same database with out having to leave the CATDADS function. The kinematic mechanism conversion takes place entirely within CATIA, and the CATDADS menu items and nomenclature maintain the "look and feel" of the CAD system. In the CATIA KINEMAT function, as joints are built ICONS appear that give reference to the boundary condition type. Once the CATDADS convert function is applied, the ICON concept is maintained, and as new elements are created additional ICONS appear on the system. The significant depth in which CATDADS is embedded is evidenced by being able to apply the CATIA LIBRARY functionality to manage action/reaction elements (tires, bushings and actuators), mechanical assemblies and sub-assemblies. This indicates that existing components can be taken "off the shelf" and mated to a larger configuration or duplicated in a different design without having to recreate the database.

The excitation input to the "assembly prototype" (CATDADS software) are external loads that actuate the system or provide resistance to component motion. The actuation may be due to a commanded motor input, pre-loaded spring cartridge, or some other action. The load resistance may be due to joint friction, aerodynamics or hydrodynamics. In future releases of CATDADS and PolyFEM component characteristics will be directly importable from PolyFEM. After the modal or flexible characteristics of a component has been determined to be important to the assembly performance assessment, the component characteristics will be retrievable from PolyFEM in the form of normal modes, static modes, and attachment modes. [4] Presently, the inclusion of the mode types is available through NASTRAN, IDEAS and ANSYS.

The system performance assessment is visualized from within CATIA. The CATIA simulation monitor is used for animating the resulting system motion. As further evidence of the CATDADS depth of integration is the usage of CATIA functionality to determine interference and collision of assembly components. The collision information can be used as feedback for modifying the design or for determining impact loads between any of the solids which would affect the resulting motion of the system. Inverse dynamics can be used to determine the system actuation requirements. This analysis type first solves for the system motion, and then determines what the required input was to enforce the desired motion. The results can be used as input to the "control/assembly prototype". The transient response and reaction loads of any component in the system can be graphically plotted to gain further insight into the system behavior. The component accelerations and reaction loads are also available as input to the "component prototyping" process (PolyFEM). Since the prototyping is taking place within the confines of CATIA the results are immediately available for upstream design processing.

5.0 CATDADS/Plant EASY5

Figure 3 - Control/Assembly Prototype Process

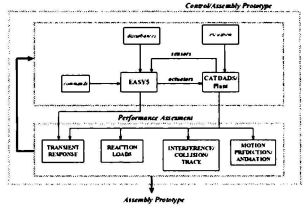


Figure 3 shows the "control/assembly prototype" process. The roots of this process are closely related to the "assembly prototype". Once the CATDADS assembly has been created, the bi-products (actuation requirements and Plant representation) are easily applied to the control design tasks. The actuation requirements can be used for the sizing of the control system, and the assembly prototypes can easily be embedded as the Plant within the control circuit. An extension of CATDADS is the Plant option. CATDADS/Plant has the capability to be automatically embedded inside of EASY5's Fortran Block, MATRIXx/ SystemBuild's User-Code Block, and Matlab/ Simulink's S-Function. EASY5 is the control software that will be applied in the application section. EASY5 is a software used to model, analyze and design complex control systems. Models are assembled from discipline-specific components, such as hydraulics, as well as primitive functional blocks, such as summers, dividers and integrators. Analysis tools include non-linear simulation, steady-state, linear analysis, control system

design, data analysis and plotting. [3] The open architecture allows the "assembly prototype" to be embedded as the Plant via the EASY5 Fortran Block. In this manner all of the control design tools are available for assessing the "control/assembly performance". The Plant model can be represented in either the non-linear or linear form. The non-linear form is that in which the resulting non-linear equations of motion are solved simultaneously with the control states. The linear form is achieved by linearizing the Plant about an operating point. And once again, due to the nature of the tool embedding, the "system prototype" may be linearized from within the control software or from the Plant software.

The manner in which the CATDADS/ Plant is readied for control system implementation is very natural. First, sensors are attached to the mechanical system. The sensors are applied to the system in the same manner as force element mounting locations are defined. The sensor types exist such that component states, relative component states and impact forces can be fed back to the controller. Secondly, actuators are attached to the mechanical system. The actuators introduce a corrective reaction load back to the plant based upon the sensory feedback and commanded input. The last step required is to set the method of integration to the appropriate control software. This action toggles CATDADS to operate as a subroutine for the respective control software. Once the control simulation is complete the results can be imported back into CATIA for visual and graphical interpretation.

The manner in which CATDADS is embedded within the control software is unique. The control states and mechanical states are appended into a single state vector and integrated forward in time simultaneously with the same integrator. In this manner actuation commands, control disturbances and external excitation are allowed to interact with the Plant in a closed-loop fashion. This leads the way to performing control structure interaction studies. The performance assessment can be further carried out to investigate how well the control affects the assembly. The depth of tool embedding allows interactive simulations to be performed which allows manual fine tuning of control gains. Since the "control/assembly prototyping" is taking place within the confines of CATIA the results are immediately available for upstream design processing.

6.0 PolyFEM

Figure 4 - Component Prototype Process

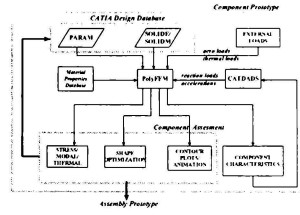


Figure 4 shows the "component prototype" process. Once again, the starting point is CATIA, and the prototyping occurs in PolyFEM. PolyFEM is a finite element analysis package that is completely embedded within the CAD package. An automatic mesher is utilized and the solution is achieved by a p-type finite element solver. PolyFEM was developed with ease of use and finite element modeling for the masses as the main focus. No expertise in finite element analysis is required to apply the technology. Since p-elements are used, features do not have to be suppressed and the mesh does not need to be manually altered to achieve first level analysis results. PolyFEM is not intended to act as a substitute for high end finite element analysis tools. The intent is to provide the means for the designer, analyst and systems engineer to gain insight into how the component will perform as the design matures. The steps required to obtain stress, thermal or modal results are as simple as fixing the part, applying the loads, material properties and selecting solve. The immediate graphical and visual feedback of results allows the engineer to determine whether the solution has converged. If convergence has not taken place, then additional solves can be initiated to achieve the desired accuracy. The first visual image the engineer sees is the component displacement which is compared with engineering judgment to validate how the loads and boundary conditions were applied.

As boundary conditions and loads are applied, ICONS appear along the selected surfaces and boundaries with different colors indicating the condition. Furthermore, the ICONS are adaptive in nature. For example, if a

load is at an angle with respect to a surface or coordinate system, the vectors emanating from the ICON reflects the angle. This further provides visual feedback to the engineer that the boundary conditions and loads were applied correctly. The input to PolyFEM is the CATIA SOLIDE/SOLIDM database and PARAM3D, external loads, material properties database, and CATDADS accelerations and reaction loads. PolyFEM only operates on the CATIA solids database without any IGES translations taking place. To assist in the component assessment part optimization, stress, thermal and modal solutions are available along with contour plots and animation's to act as graphical and visual aids to the engineering judgment process.

7.0 Application Example

Figure 5 - Landing Gear Drop

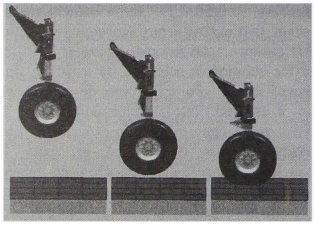


Table 1 - Landing Gear Drop Test Values

Test No.	Weight (lbf)	Vertical Sink Speed (ft/sec)	Landing Air Speed (mph)
1	40000	5.0	150.0
2	40000	10.0	150.0

The landing gear shown in Figure 5 is a replica of an F-15 Eagle main gear. The three images show the gear at three different stages of a drop test. The load case for part one of the application example is a simulated drop test. The variables for the load case are aircraft weight, vertical sink speed, and landing airspeed. Table 1 shows the variable values for the two drop tests. Starting with SOLIDE geometry and an incomplete KINEMAT set inside of CATIA, the kinematic model is converted to a dynamics model using the CATDADS functionality. The dynamic model has 3 DOF: tire rotation, oleo shock translation, aircraft vertical motion. Figure 6a shows the oleo strut force, and Figure6b shows the planar load on the lower torque arm. The transient responses were obtained using both sets of test conditions shown in Table 1. The interference and collision features of CATDADS were also used to insure that no obstructions developed during the drop tests.

Figure 6a - Oleo Strut Force

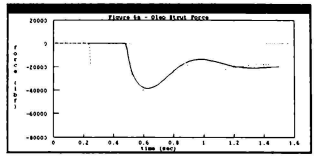


Figure 6b - Planar Load

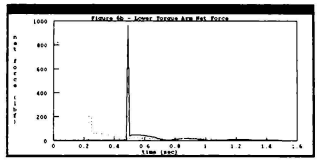
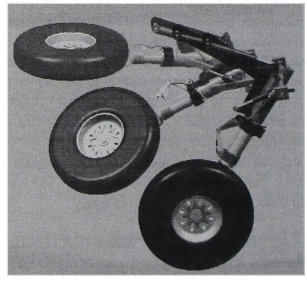


Figure 7 - Landing Gear Retraction



The second part of the application example involves fixing the aircraft's vertical motion and releasing the retraction actuator such that the system still has 3 DOF. Figure 7 shows the gear undergoing retraction. The CATDADS/Plant EASY5 extension was used for simulating the retraction hydraulics. The hydraulic circuit is shown in Figure 8. From within CATDADS, control sensors were connected to the retraction axis for feeding back the stroke, velocity and acceleration to the controller. An actuator was also connected to the

retraction axis for applying the output of the hydraulic circuit to the gear.

Figure 8 EASY5 Hydraulic Retraction

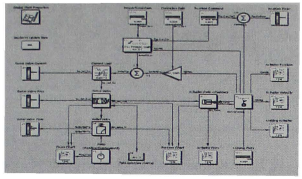
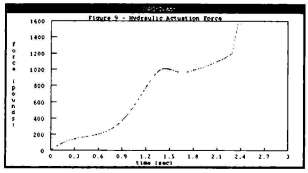


Figure 9 shows the actuation force required to retract the gear.

Figure 9 - Hydraulic Actuation Force



The third part of the application example involves applying the peak load shown in Figure 6b to the lower torque arm shown in Figure 10. The part was fixed, loaded and a stress solution was performed. The same part was re-evaluated with a web and fillet. The deformed and undeformed lower torque arm is shown in Figure 11 before and after the redesign. The deflection and stress summary is shown in Table 2.

Figure 10 - Landing Gear Lower Torque Arm

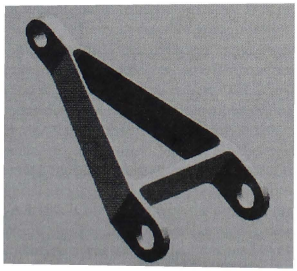


Figure 11 - Lower Torque Arm

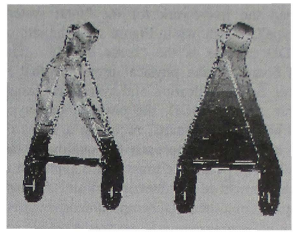


Table 2 - Deflection and Stress Summary

Torque Arm	Deflection (in)	Stress (ksi)
before	0.070	100.0
after	0.011	35.0

8.0 Summary

The SDD process applies to all types of systems encompassing kinematic, component and controlled motion design. The paper has presented "CAD embedded CAE tools". The CAD tool applied was CATIA, and the CAE tools were CATDADS, EASY5 and PolyFEM. The application example was an electronically prototyped aircraft landing gear exposed to a drop test and retraction simulation. The performance assessment resulted in the CAD database being naturally updated to achieve a more efficient design. The efficiency was achieved by operating from a single CAD database with the results being available for upstream processing. With this approach, training requirements and the number of experts are significantly reduced which suggests that virtually all related engineering establishments can apply this technology without having to maintain a high level of readiness to apply the CAE tools when the need arises.

SDD brings PDT to their full fruition. The teams, designed and configured to eliminate the "over the wall" approach to design, still have inter-team barriers due to the need to translate data from the design database to "integrated" CAE tools. The "integrated" CAE tool approach contributes more non-value added activity and fractures the system, design and analysis databases. The SDD approach facilitates the

mission of PDTs by merging the system design and analysis databases. During the process of applying the SDD, the CAD environment provided the framework for the "total system prototype". As shown in Figure 1, at the end of the SDD process is the "brass board". The "brass board" is the physical prototype that is used for final verification and system checkout. Without applying SDD, the physical prototype would be a "bread board" requiring a series of costly iterations to approach the quality of the "brass board". Thus the brass board approach is the achievement of the "new paradigm" through the application of the SDD environment.

References

1. E. J. Haug, Computer Aided Kinematics and Dynamics of Mechanical Systems, Volume 1: Basic Methods. Needham Heights, Massachusetts: Allyn and Bacon (1989).
2. Anon., *PolyFEM User's Guide*, Computer Aided Design Software Inc., 1997.
3. Anon., *EASY5 User's Guide*, The Boeing Company., 1997.
4. S. Wu, E. J. Haug, and S. Kim, "A Variational Approach to Dynamics of Flexible Multibody Systems," *Mechanics of Structures and Machines*, Vol. 17, No. 1, 1989, pp. 3-32.

THE BOEING COMMERCIAL AIRPLANE ENGINEERING SIMULATION, NEW AIRPLANE PROJECT, MANAGEMENT OBSERVATIONS

George B. Williams
Principal Real Time Simulation Engineer
The Boeing Company
P.O. Box 3707, MS 19-MH
Seattle, WA 98124
(206) 662-4755

Abstract

The real time engineering commercial aircraft simulation begins as a minimally functioning tool, demanding less than high fidelity system functionality, for specific limited engineering tasks. It is expected to continuously evolve, improve, change, become more and more like the aircraft, be under configuration control and released in a coordinated fashion. It should be easy to use, flexible enough to be configured with any perceived hardware test setups, execute in a real time step size which doesn't compromise control law fidelity or exceed any data latency limits, and interface with all necessary hardware (aircraft hardware LRUs, simulated flight decks, input, output and host computer systems). Yet, it must finish as a high fidelity, full system functional, pilot procedure oriented, Interface Control Drawing (ICD) compatible, ready for flight test data reduction simulation tool.

Introduction

This is a summary of management observations of an airplane engineering real time simulation development project. It is based upon the 777 program and previous Boeing airplane programs. It presents a line management perspective of difficulties, values and key issues associated with this type of effort.

At Boeing, the task of developing, integrating and supporting the aircraft real time simulations is performed at the Integrated Aircraft Systems Lab, IASL. This lab provides this product/service for all Boeing commercial aircraft programs. Our "customers" are the aircraft designers, testers, trainers and vendors.

Many issues impact the success of the simulation project. The different and numerous dimensions can be overwhelming. The roles and responsibilities of the lab and program personnel vary with the customer and the application. The degree of coordination between customer groups varies from non-existent to excellent to "too much". Different customer groups/design technologies have different goals and requirements. The engineering methodologies vary. The customer desired and provided lab services vary. The schedules for these services and simulation functions differ. The experience of the customers with lab and simulation development processes vary. Most of all, the overall management expectations and understanding of the lab deliverables, services, and role in the airplane program vary. When this is combined with the airplane program perturbations, facility moves, new personnel, new management, re-organizations, and the need to establish budgets and software development schedules, it is quite an experience.

Although significant to the project, the executive software, utilities and hardware will not be addressed here. The focus of this paper is on the aircraft system model development, integration and application of these models within an already existing yet changing development environment.

People

There are several types of simulation work performed in the lab. There is the initial work to develop and integrate model simulations. There is the effort to provide standard, tested, documented, released simulation elements and configurations. Most importantly, there is the perpetual ongoing application of these simulations in testing and evaluation of designs and handling qualities. All these activities are accompanied by a degree of discovery and require a team of experienced, capable and motivated engineers.

These people are the most important part of the project. Almost any deficiency in the project can be overcome if the team has the right people. The Boeing simulation engineering team has become one of the key contributors in the process of designing and testing aircraft. It can be said that, today, without this function, the task of developing and producing a new commercial aircraft could not be done.

From the era of the SST to the 777, Boeing simulation technology has been growing and evolving, demanding a broad range of skills and expertise. It relies upon a special relationship between aircraft designers, computer system specialists, software and hardware designers, modelers, experienced real time engineers and knowledgeable management. The people have

extensive knowledge of existing simulation systems, aircraft systems design and operation, the ability, and the expectation to meet unanticipated needs.

These professionals bring with them an understanding of the basics of aircraft simulation, the underlying requirements, the historical problems, the applications, and associated engineering/software processes. This is recognized in the company as a critical technology, and valued as a core competency.

Together, the many lab customers and the lab simulation software engineers construct the best real time commercial aircraft engineering simulations on the planet. They do this concurrently with the aircraft design. They complement and rely upon each other's processes. They learn from and teach one another.

Customers.

There are multiple perspectives on who are the principle customers of the lab engineering simulation. This has been one source of contention when trying to place the lab in an organizational context. The lab affects the outcome of many different organizations

Four major groups of simulation customer exist. Two are internal to the company, the aircraft designers, and the testers. Two are external, the LRU vendors and the crew training simulator builders. The flight test pilots are part of all the internal engineering activities. The LRU vendors are a recent addition.

As a key element of the aircraft design and analysis, the simulation would exist if only for the design applications. However, testing ("flying") the avionics, on a bench, in a controlled and repeatable environment has considerably benefited the on time delivery of service ready aircraft. (Thus the avionics view of testing as the principle requirement for the engineering simulation) Crew training requirements demand simulations, model specifications, and support data and have been a major driver of increases in fidelity.

From a chronological point of view the aircraft designers are the initial customers for and developers of the engineering simulator. Other users/customers benefit after the fact and impose additional, sometimes unanticipated requirements.

At first, minimal fidelity is required. The ability to trim the aircraft, set and drive control surfaces and balance drag with thrust is all that is needed. Software models from previous programs usually provide the base of this development. Models are copied and used after some necessary modifications. The Stability and Control engineers then arrive with their wind tunnel data.

Following the Aero activity comes the Control Systems , Mechanical Systems and Propulsion model development. Given the availability of an accurate set of aero, propulsion, and control system model, the auto flight engineers can then perform analysis and refinement of their preliminary designs. On the 777 program, as with each new program, additional modeled systems were required and existing systems were elaborated.

Finally, come the LRU bench testers and the aircraft System Integration Lab (SIL) testers. After the design and construction of the hardware test facilities, the simulation elements are configured and integrated with LRUs, actuators, etc. and "flown". Thus begins a series of evaluations of hardware and simulated systems, with continuous update and change cycles.

Throughout all of this, Flight Deck engineers want all the systems simulated as early as possible. As an early simulation user, Flight Deck engineers have the most demanding of fidelity requirements in support of cab evaluation and pilot procedural testing. (They are also of the opinion that the cab, a functioning flight deck mockup, is their domain and reluctantly suffer its use for other engineering purposes.) They need to have the simulation checked out, functioning perfectly, in a cab mockup with a complete feel system, on line, in advance of any complete aircraft design. The earliest dates demanding the most function of the 777 engineering simulation were those of the Flight Deck customer for a flight ready engineering test cab.

A key difference between the Flight Deck customer and all other company internal customers is that they do not provide model designs. They bring requirements. This is a major challenge to the lab simulation engineer in that many of the requirements are not satisfied by the existence of a simulation/aircraft design activity. That is, for some of the systems needed by flight deck, there is no system design activity in the lab (via lab real time simulation).

This puts the simulation engineers in the position of having to determine or research system specifications. New requirements of simulation models/systems are discovered, often after manpower budgets have already been established. Flight Deck always requires unique support from the lab, with schedule-compressing needs.

Although simulations have existed for quite some time, not all the aircraft system design processes involve the use of the IASL engineering real time simulations. There are always some engineers and management who newly discover the power of the real time

simulator and apply it to problems and questions previously dealt with via their own local efforts.

As the advantages offered by higher fidelity simulations become evident, the number of simulation users/customers and applications continue to expand. This presents challenges when some do not understand the cost increase or the return on investment. Simulation customers design methods benefit but the cost of the simulation effort increases. We believe (but have yet to collect data) that the net cost is reduced. The complete view of the advantages and the applications of the engineering simulator is not in everyone's experience.

The LRU vendor community is and will continue to avail itself of the engineering simulation. On the 777 program, a Boeing vendor was provided the engineering sim, with updates, to use in their testing. It has been questioned whether this testing was best accomplished with the high fidelity sim or whether a simpler tool would have been better. As the complexity of the engineering sim increases, so does the difficulty of use, the need for more detailed understanding of the tool as well as knowledge of the operation of the aircraft systems themselves. Chief engineers have stated the desire to have vendors do more testing of the boxes. It is anticipated that providing vendors with the engineering simulation will continue with ever greater application, benefit and cost.

An indirect customer of the simulation effort is the crew training community. Almost all of the major model specifications for the crew training simulators are developed and implemented in the engineering simulator. With increasing demands of airlines and the FAA, the simulator fidelity and training requirements have expanded and imposed upon the engineering simulation development schedules and aircraft development schedules. Better and earlier training is required. To meet training simulator on-deck dates, simulation specifications must be provided in time for development, delivery and acceptance of the training systems. Updates to predicted simulations are scheduled as well as proof of match, based upon flight test data. With the pursuit of shorter development cycle times and the desire to get airplanes built and delivered to customers faster, the engineering schedule is experiencing a bit of a pile-up. The engineering simulation schedule is being driven by crew training needs as fast as the design schedule itself.

The customer's skill and experience in engineering real time simulation development and use are many and varied. Their roles and responsibilities are not often common across the different technologies or agreed upon. From the labs view, the customers are responsible for defining the aircraft systems (and simulation specifications). Most of all, they are responsible for approval (check out and release authorization) of a particular simulated system for engineering use by other aircraft designers. Further they are responsible for assessing the proper integration and function of all the pieces, the total simulation.

Some of our simulation customers, with their own design focus, are not always aware of (or possibly concerned with) the simulation related issues or the questions and compromises posed to other dependent aircraft design and test activities.

Company directives(that the various design technologies support the simulation effort) are only as successful as the design customers ability to invoke and monitor that support. The customers must communicate their requirements to the simulation engineers as well as to one another. The role of the customer is to establish mutually agreed upon requirements and dates by which model specifications shall be provided to the simulation development organization. The customers and the simulation development organization must then negotiate schedules such that the need dates are known and met. This is a key customer role. Without this, the simulation and airplane schedules are at risk.

In some instances, lab customers have assigned specific individuals with the responsibility for all their organization's simulation related activities. This has proved to be a very beneficial approach. It allows focus on the implementation, checkout, problem resolution, lab activities, management of the specifications, and schedules such that program milestones are supported. The simulation effort benefits.

Other customer organizations have combined the role of designer with that of the simulation responsibility. All the aircraft design and test activity, in addition to the simulation related work can cause conflicts in priorities and delay simulation development or function. (Documentation is often their first casualty followed by integration testing). The pressure to fill in a scheduled milestone while minimally meeting the criteria can become overwhelming.

Customer engineers depend upon the simulation to evaluate aircraft design. They work directly with simulation software engineers providing model block diagrams and algorithms to be implemented and checked out. Some of these customers need these activities in advance of other aircraft system design development. Some need the results of other development so that they can proceed with their design

and evaluation tasks. As with everything the degree of experience of a customer, their accepted role and responsibility assists or hinders a new program.

Simulation Personnel

The customers have many different expectations and needs of their assigned simulation development and application engineers. These applications engineers are expected to:

a) be responsible for the design and implementation of the model specifications,

b) maintain configuration control and track all change to their model,

c) assure that the developmental version of the model makes it into the released integrated simulation with no unapproved changes or unacceptable problems,

d) have the knowledge of the integrated simulation, associated tools and utilities,

e) be the local, available, on site experts in the area of their model (simulated aircraft system),

f) help train their customers (as required),

g) know the simulation development and application processes and assist their customer in using them,

h) know the re-useable software and its application,

i) understand real time issues and the required coding standards and practices,

j) be the customer's advocate in the simulation facility.

Finally, the simulation application engineer is expected to assist their customer, and any other lab engineer or customer in the use and application of the simulation. This can be in a real time environment with a cab and certain LRU components, or it can be in the customers or their fellow engineers' "workstation" environment in their office area. The simulation engineer knows or determines the options with respect to testing techniques and capabilities. They also develop test scripts and automation to facilitate accurately repeating tests and the generation of documentation. The lab simulation application engineer is the "tool" expert and the on site (in the lab) model expert.

Being the local expert in a particular model/system area also requires the lab applications engineer/modeler to be available for consultation and support of any simulation applications requiring their model knowledge. One can be on call whenever any testing difficulty presents itself, either test set up or in conducting the test, in the bench environment or cab or office work station, whatever is required. As such, given the lab can run 24 hours a day, the engineer can expect to participate at any time of day or night if the situation calls for their expertise.

To help with this expectation, the 777 program was the first new airplane simulation program to be conducted under Variable Work Schedules, (VWS). This was previously a successful pilot program within the IASL 737 lab support organization.

The simple concept of VWS was that the people doing the work know best how and when to do their work. Let them schedule their work.

It was a voluntary pilot project and almost all participated. Individuals were required to post weekly schedules at their desks, approved in advance but not unchangeable. They were expected to attend all scheduled meetings. If their work was accomplished early, fulfilling obligations and the eighty hour criteria (or more in a two week period), an occasional day "off" was possible.

Given the complete and absolute success of this pilot, and no retrogression in management philosophy is realized, this work approach is here to stay. Employees and line management have claimed reduced unnecessary overtime, more flexibility in meeting customer needs and a greater ability to meet changing program commitments. Most of all, we have had happier and more effective employees.

Paraphrasing one high level manager in an all hands 777 program meeting, "This is our profession, it's what we wanted, it's what we trained for, and it doesn't get any better than this."

The lab engineers who support simulation modeling are computer scientists that develop aircraft simulations as well as aircraft knowledgeable people who write software. They are a technology in and of themselves. They do not build aircraft. They design, build, and apply aircraft simulations. Many degrees, experiences and backgrounds exist. The most successful way to view (to manage) these people is as a group, a team with a spectrum of skills and experience. Over emphasis in any area will dilute the ability of the overall team. With a diverse team, problems can be understood and solved more quickly, missing requirements and implementation details can be better tolerated and resolved, and communication with the customer engineer can be greatly enhanced. Lab engineers have recourse within their own organization to seek and obtain help in modeled areas. As long as each engineer in the team, with his or her particular skills, has the willingness to assist one another in

whatever capacity, with the desire to succeed, the best of all possible worlds exists.

Getting the right people is the most important step in the project. The 777 engineering simulation software team was among the best. Putting this team together was the first success of the program.

The Lab

The 777 lab project had an excellent management team. It had to. It had about as many difficulties as one would want.

For the 777, previous lab development processes were not adequate. With new management and new company orientation, topics such as communication, lab direction (mission), methodology, different business processes, and different expectations were important and necessary, yet often unresolved.

The 777 program evolved into existence beginning with several "lead in" study programs, the 7J7, the 7X7 and the evaluation of the new control system approaches with the 757 test bed.

Prior to the 777 program, the simulation lab existed in the corner of a mockup building in the Renton complex. A new lab facility, the Integrated Aircraft Systems Lab, (IASL), was constructed but not before the program was already underway.

While supporting the new airplane program, a wholesale move of the entire lab was necessary. All computer workstations, networks and host computers, test benches and engineering cabs were moved. The former facility served only the cab functions during the transition.

While this served as a distraction, the lab operations people were more than successful in planning and executing this effort. Little negative impact resulted for the 777 software development organization.

In addition to the new lab, new real time computer platforms, different engineering workstations (double the software support please) and new multi processor executive software and utilities were developed and in use.

Projections and estimates of the lab part of the 777 program grew beyond all previous program levels. New organizations were created out of the lab organizations supporting the preliminary study efforts. These were primarily the hardware test support organizations, continuing to wrap up the previous activities but postured for the new 777 airplane program.

The last and final 777 lab organization to be put in place, prior to the lab move but after the program go ahead, was the simulation software organization. It was started late, wasn't completely staffed, had little budget established, and much of the staff had to come out of existing multiple lab organizations.

In the beginning of the 777 program, the lab support functions were organized per specific customer organization, combining lab test hardware support and the simulation software development. When the decision was made to collect all the simulation software activities into one organization, not all moved. At least one function stayed in the customer oriented mode. This caused problems. Given the admonition that normal software projects always take longer than planned, more so for an engineering simulation, this caused much stress and pain for line management and the engineers.

With a late start and after a year of trying to catch up, more effort was placed in the 777 simulation software organization. Another reorganization occurred. Although this was the source of yet another set of expectations and views on methods and procedures, this help (change) was a net improvement for the simulation software project. The original budget goals were met, the schedule milestones were filled in, the "suitable for framing" awards were awarded, and plans for the derivative begun.

The Simulation

There are at least four major phases of simulation activity for new airplane programs: preliminary studies, the airplane model development/aircraft design phase, the testing phase, the flight test update phase. Of course they don't all occur at the same time for all systems, nor are they neatly identifiable and bounded by start/stop dates. Change always occurs as with the aircraft design. Throughout the process, change requests and problem reports abound, as well as newly identified requirements which weren't budgeted.

The simulation organization is active throughout all phases of the program. When the 777 was finally formed, it incorporated the organizational approach of previous efforts with respect to models and their inherent relationships. The models which made up the basic components were together, under one lead, and the Primary Flight Control Computer (PFC), avionics and flight management, AIMS and Cabin Systems were under their own lead engineers. To this common organization was added three additional groups, software integration and release, a tools group, and the project planning perspective, all under one first line

supervisor. Other lab organizations supported the simulation organization in the area of computers, operations and system software.

One goal of the Integration and Release group was to establish a standard simulation configuration, the Integrated Aircraft Configuration (IAC). The IAC was a fully integrated, configuration controlled, checked out simulation comprised of all software models. This would be the baseline against which all other test support simulations, configured differently, would be compared. It would also provide the standard from which crew training data documents (static data and time histories) would be generated. All unique model checkout data would be within the same released context. This basis is highly valued when pursuing crew training vendor implementation questions later. This goal was achieved.

The singular position of the project engineer was developed to facilitate customer and lab communication.

Most of the customers and lab engineers were already familiar with simulation, with the right skills. They were also up to speed on the new equipment and real time system executive and utilities.

The reality of new airplane programs is the need for ever greater computer performance and capacity. The simulation development environment, executives, platforms, utilities, tools, and associated software and processes should be established prior to and in anticipation of the coming applications. This was the case for the 777 program. The lab was prepared in that respect.

The simulations have continually grown in size, complexity, fidelity and scope of application. With each new program more aircraft systems are included in the simulation. Aircraft have more systems. The fidelity of the typically simulated systems has increased beyond that of previous programs. With the increasing complexity of the aircraft systems, so goes the simulation requirements and resultant complexity.

The design of the simulation is basically the same for all the commercial aircraft. The functional elements , the relationships, the data dependencies and flow are all the same. With each new airplane, additions have been made to attain greater and greater accuracy in the models and the predictions of airplane characteristics. The increase in size and number of the models is causing problems. New avionics technology is causing complexity. The questions which we want the simulations to answer are expanding in scope and number. The simulation is the only viable tool to answer design questions.

To optimize aircraft design, to properly ascertain the function and interaction of the aircraft systems, to see design flaws and rectify them before metal is cut or specifications finalized and implemented, for each design organization to mutually benefit from each others' expertise and model, to "pre-integrate" the aircraft, this is the value of the simulation.

The marriage of the real time simulation effort with the airplane design is here to stay. The advantages of the individually developed stand alone design simulations are no longer cost effective. They can no longer adequately test a given system. With the greater complexity of aircraft designs, high fidelity integrated systems testing is required to properly evaluate their performance. The modern commercial aircraft engineering simulation is absolutely necessary in the development process of new aircraft. The only thing better than today's real time engineering simulation is a full System Integration Lab (SIL) or the aircraft itself.

Simulation Development

The practice in the development of the simulated systems (or models) is to team the lab engineers with customer engineers in each specific aircraft system. The customers provide the aircraft system knowledge and the lab engineers provide the specific simulation expertise. With time, the lab engineers come to know the specific aircraft system details and function as do the customer engineers with the simulation tool operation and functions. They gain knowledge of each others perspective.

In the initial stages of development, the lab engineers may reside on site with the aircraft design customer. This may involve the implementation of control laws, from markups of previous aircraft design documents, the development of model algorithms and associated functional data bases. It may further cover the initial customer training and familiarization with the simulation software, utilities and tools.

Given the necessity and complexity of the engineering simulation and the key part it plays in the airplane design and test, it demands planning and control as much, or more, as any other project. The simulation modeling and system design development methodology, however, presents some difficulties.

One of the most challenging aspects of the typical Boeing engineering simulation project is the dynamic and evolutionary nature. The simulation is not developed from a formal specification, but in real time

by teams of customer and lab simulation engineers (typically per simulated system). The simulation is built in functional layers and steps. Over time these individual system developments are integrated and released to a continuously growing group of simulation developers and users. The cycle repeats. On day one, the simulation is nonexistent. It is needed initially to evaluate the aircraft aerodynamic design and handling qualities. Those results are "released" to others who use it to do their system studies and designs. Those results (model updates) are in turn released to the engineering community and so on with ever increasing fidelity and function.

The important part is to have functionality available when needed by a given system design customer. At certain stages, all development has a functional relationship/dependency upon the other simulated systems. The various simulation functions are dependent upon the state of the aircraft system design and schedule. The individual customers know their design plans and dates. It may or may not be the case, that the dependency of these dates upon the existence of a released engineering simulation is known by all concerned parties. This information must be actively coordinated, simulation delivery dates and development schedules negotiated.

Before any released simulation exists, there is a minimum set of simulation functions which must be developed. This is the most difficult phase of the project, the initial development and integration. This phase is dependent upon the existence of related functions and a common interface.

In the initial stages of the 777 customers continued to use a previously established prototype simulation. This simulation existed in support of the PFC 757 test bed. Design engineers had confidence in this simulation. It had been used for some time in advance of the 777 go ahead. While this usage was beneficial to system design, it forestalled specific 777 simulation model development and checkout.

There were other difficulties. The functionality required for Flight Deck 777 cab studies was not available per program schedule. The solution to this problem was a change in the typical role of the lab.

The lab ultimately took action and became the broker for the design community. A series of all customer meetings was held to establish and communicate functionality requirements and completion dates for implementation across all simulation models. This was the single most important lab event in the 777 engineering simulation development.

This activity was necessary and one which exceeded the historical role of the lab.

Program dates for delivery of airplane system specifications (SCDs) to LRU and crew training vendors typically existed, but pre-implementation in the engineering simulation was not a criteria imposed upon this information.

After lab action to establish minimum functionality requirements and obtain program commitments to provide model specifications to the lab a plan for first release with known functionality was established.

A forum to communicate crew training needs and dates (for documentation) had previously existed. It later was expanded to also help monitor and guide the customer simulation model development, checkout and schedule commitments. This was a new organization called Integrated Simulation Management, (ISM).

The simulation effort is directly linked to and dependent upon the design customer processes. It is the result of their design development studies which ultimately are incorporated into the "released" engineering simulation and used by the rest of the company. Our "customers" work with specific groups of lab engineers developing specific aircraft system design simulation elements. They do studies and analysis and the lab implements their algorithms and integrates the developmental software with the current released versions. Once this software is at a major functioning point, it is released. If these customer designers have difficulty such that it is not coming together, the simulation reflects that. If they do not realize requirements in a timely fashion, the simulation effort is affected.

Conflicts exist between aspects of aircraft design, the development of the simulation, and the need to establish software development schedules with specific goals.

Given the lab's customer/supplier orientation with the program, how is the sim schedule developed and imposed upon the design customer? The ownership of such tasks and the monitoring of such tasks is an important issue.

It is always the case that the simulation requirements of one aircraft design organization imposes upon the development schedule of another aircraft design organization. For the typical experienced simulation users/aircraft designers, these impositions (requirements) are known and coordinated between customers.

GSDS

One tool used to develop the simulation models is the Graphical Simulation Development System (GSDS). It is an icon based functional block diagram generation tool. Ready to compile code can be generated from the output of this tool. The single most valuable benefit of GSDS is that it also allows many or most of the elements of the LRU specification and the Crew Training model document to be common with the simulation. This has all but eliminated redundant steps and the possible disagreement between simulations, results and other specification documentation.

This tool is used by customer engineers as well as lab engineers to develop the simulation models. This can cause problems. Customers do not always know the lab standards and conventions. The results do not include some of the simulation requirements for simulation initialization and mode control and real time execution considerations. Also, when the model is composed without the participation of the lab engineer, that person has less knowledge of the function and operation of the model and can not completely fill the "local expert" role.

For the experienced customer, this problem has been minimal. For the new customer, with no experience or understanding of the bigger picture, this can be expensive. For at least one of the 777 simulation models, the delivery of the GSDS generated information was always followed by some lab activity so that it could work with the other models and the simulation. The same activity, (redo and integrate), had to be performed every time there was a customer update to the model. Also, lab implemented changes, had to be re-implemented in the customer generated model. Configuration control, coordination and accommodation in this instance were less than desired.

Given the benefits of this tool, the drawbacks are tolerable and addressable with other process modifications.

GSDS holds great promise. Given the definition of the models in this context, the possibility exists for the Crew Training vendors and perhaps the LRU manufacturers to directly apply the code generation. Their current costs could be considerably reduced but problems similar to those of the lab engineer and our customer generated implementations might exist.

Budget

After the people, the budget is the most important part of the project. Experience has shown that the basic design customers usually have the ability to establish and provide the lab with budget. They know who is impacted if insufficient lab resources exist. These individual budgets are in the form of engineering work authorizations.

The 777 simulation software budget exercise eventually had the participation of high level management from both the customers and the lab. The avionics estimate was based entirely upon what are called One Page Test Plans, (OPTPs). These plans often identify high-level needs for software simulation development but don't always directly state the detailed requirements. When they don't, the simulation organization is responsible for working with customers and reaching an understanding on the magnitude and establishing workable estimates.

PFOD

Additional management was added to the software effort with a reorganization after the initial release schedule slide.

A successful effort was made to establish a known agreed upon, published schedule of simulation software IAC releases throughout the entire program. The new release schedules were to occur in conjunction with the newly established Proven Functional Operability Dates (PFOD). Releases of the airplane ICD, the data base of the Interface Control Drawing, would coincide. The development of the simulation after the first release was based upon documented problem reports and change requests, typically dealing with the evolving functionality.

During the 777, the simulation software functionality was often leading ICD and Specification Control Drawing (SCDs to LRU vendors) updates, as driven by customer delivery commitment dates. The simulation is where many of these specs get developed. Not all functionality implied by ICD updates was required or necessary in the simulation. Unfortunately lab based LRU testing was based upon older released versions of the modeled systems. LRU vendors were two releases behind in their ability to stay in step. Flight Deck wanted the latest simulation function available for cab testing (even using beta released software).

At least two releases, thought to be needed by the LRU vendor could not be used by them. They had yet to fully bring up the first delivered software. The lab simulation software engineers had to support and update old releases of simulation models. This was required because of the functional and interface incompatibility of the current designs (the simulation elements and interface). There was a lag in the flow of simulation/SCD development to LRU delivery and test.

This schedule compression and requirements overlap must be anticipated in the future with accommodations in place for phased hardware delivery and continued support of compatible older simulations. Each simulation release has its own life cycle.

IAC Only

With the reorganization of the software effort, the emphasis on the IAC was all encompassing. The IAC concept was accepted but the need was recognized not to address individual software issues. Planning only covered the IAC and did not recognize individual customer model development schedules. Customer problems do not always occur in concert with major release plans but accommodations for other than IAC software releases were needed. Single model releases and software fixes, within a configuration control context, were also necessary in the lab test environment. These processes were ultimately put in place.

Company Initiatives and Change

Prior to and during the 777 program the company was pursuing many initiatives and management/employee orientations and training. "World Class Competitiveness" was a frequently heard term, "Continuous Quality Improvement" another. All were expected to look at the as is process and improve it. Many instructional activities ensued. DTA/BPA (Detailed Task Analysis and Business Process Analysis) was to be performed. Following that was Work Management. All of these had the promise of positive results. One particularly noteworthy initiative was the advocacy of teams.

An exceptionally successful team, critical to the 777 simulation project was the ADT, Applications Design Team. In this team the many different simulation design issues were dealt with. They were made visible to all and standards and guidelines resulted. During the 777 program it applied only to the 777. This team continues today and has been expanded to cover all aircraft engineering simulations. It is the primary lab process for approving common designs and methods in the simulation software organization.

Another organization which was established late in the program was the Integrated Simulation Management team (ISM). It was a customer chartered team which watched out for simulation document delivery dates and served as a single voice (when required) for the customer design community. The simulator documents relied upon the implementation and check out of the models in the engineering simulator. A goal of the program, was to provide a coherent and coordinated set of simulator design and checkout documents based upon the same specific released engineering simulation. The advantage of this, obviously, was the common basis across systems. All generated simulator data would be with respect to the same data and functionality. No differences resulting from independently developed crew training models would exist. Later in the schedule the simulator vendor questions on their crew training simulator results would be easier to address and within a known context. This approach was in place on the 737-400/500 programs, but this was the first new airplane program to achieve this result. This method will undoubtedly be used on future programs.

One function which was not within the charter of this ISM was that of monitoring the development of the model functionality. The focus was on the effort to meet simulator document schedules and not program design and testing schedules. The simulation model development responsibilities were added later. They helped prioritize work when release IAC dates imposed deadlines.

Summary

With each new airplane comes an order of magnitude change in fidelity, application, size & number of simulated aircraft systems, customers, users, management, and tool complexity. This has required the latest in computer platform performance.

To arrive at a simulation development plan, a detailed set of system functional requirements and the dates by which specifications are finally delivered to the lab must exist. The dates for delivery of those specifications to the lab must be visible and monitored at the airplane project level.

The effort to develop the simulation benefits from experienced manpower. It requires changes in methods and processes, not necessarily perceived in advance of the program. Most often, these changes must be dealt with and resolved concurrently with the program.

The process of commercial aircraft design substantially benefits from and is entirely dependent upon high fidelity real time engineering simulations.

Roles and responsibilities of all participants in the engineering simulation must be clearly defined and agreed upon. With each program and management team, they change. It may very well be that the lab is now expected to dictate model specification delivery dates to the design customer community and monitor those activities.

This is no grand new realization but important enough to state. Get started early. Do not wait to start the software project such that shortcuts become attractive, and planning and control steps are missed or out of order. The organization must be established and staffed as soon as possible.

The Graphical Simulation Development System (GSDS) provides much improved throughput for aircraft design processes. This may increase the effort of the simulation organization, as customer generated simulation design may cause integration problems, may not align with lab standards, may only meet the narrow view of that customer's requirements, may not meet minimum generic requirements, and may imposes rework on the simulation designer and integrator.

A customer organization, working across all technology simulation customers, must exist. They can monitor lab customers obligations and activities, conduct coordination meetings, impose due dates, resolve requirement conflicts, field unanticipated new requirements, and be the owner of the program simulation related milestones. It is important that they are appropriately positioned in the organization so they have the clout and authority to direct "lab customers" and "lab suppliers", establish priorities and obtain program commitments.

With each completely new airplane program, significant changes in processes, platforms, customers, roles and responsibilities, management and personnel should be anticipated.

A full spectrum of skills is required of the simulation engineering team. Given the simulation engineers skills are correctly assessed, properly matched with their tasks, and given the authority and responsibility, they can make it happen in spite of the difficulties.

Variable work schedules (VWS) were in place for the entire 777 program for the entire lab. The ability of the simulation engineer to work on a variable schedule greatly enhances their morale, productivity, and reduces the cost of the project (opinion).

The 777 simulation software project was very successful. The project was within budget and received high praise for it's overall success. Many of the processes established during this program have been adopted and applied across the whole of the simulation organization.

AIAA-97-3795

MODERN COMPONENT METHODS FOR AIR TRANSPORT ENGINEERING SIMULATION SOFTWARE

William H. Walker and Robert C. Kircher Jr.
The Boeing Company
PO Box 3707, MS 19-MH
Seattle, WA 98124-4404
206-662-4928 and 206-662-4962

Abstract

The size and complexity of Boeing air transport engineering simulations has grown dramatically over the past two decades. Coincident with this growth has been the equally dramatic increases in computer power. Unfortunately, even with the increased computing power, the time required to rebuild a 777 simulation has grown unacceptably long. Thus, the necessarily iterative edit-build-test cycle of simulation development to support the design of modern control systems becomes increasingly inefficient. Similar scenarios are being played out in non-simulation computing arenas such as enterprise business computing applications. The UNIX environment provides simulation software developers a rich toolset for the development of customized component architectures. Boeing's Airplane System Laboratory (ASL) has developed a novel method to componentize existing 777 engineering simulations. The result has dramatically reduced build times with minimal side effects. This paper shows the evolution over the past two decades of the size and complexity of ASL's engineering simulations. We describe in detail ASL's novel approach for reducing simulation build time, and show the resulting software process cycle time benefits. Additionally, we provide an overview summary of the computing industry standard CORBA and DCOM frameworks and suggest applications of the DCOM framework to the air transport simulation development arena.

Introduction

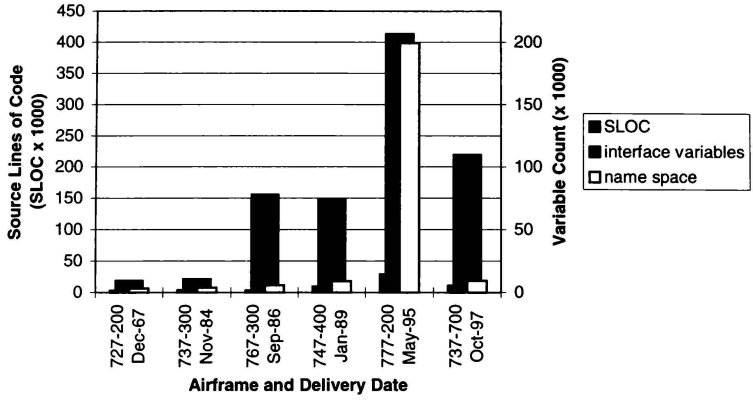
The Boeing 777 development program brought dramatic increases in both size and complexity to simulation model development. These increases caused classic scaling problems in existing approaches and solutions. As early iterations of the 777 simulation were being built it became obvious that a much larger percentage of our time was being spent on process rather than development. The cycle time for stabilizing model interfaces was days or sometimes weeks. The magnitude of interface changes precluded unit testing and resulted in days of additional integration testing to trace the source of differing results. To construct the monolithic 777 simulation required a minimum of forty minutes and took over two hours for some of the test bench configurations. All the while the model change requests were piling up, the simulation was not meeting frame time requirements, and the 777 scheduled delivery date was marching closer.

The Boeing Company relies on simulation software for commercial airplane avionics development. The simulations are used to define requirements, test hardware, refine flight deck designs, and test flight characteristics prior to actual flight. Three factors have contributed to the dramatic growth in size of airplane simulations, the first being the emergence of digital avionics systems. In 1980, the few simulations that existed were used only for "basic" airplane characteristics. As digital avionics were introduced in 1981 on the 757, 767, and 737-300, simulations of Autopilot and Flight Management Computers doubled the size of the basic airplane simulation. The development of the 747-400 in 1984 again doubled the size of the simulation with additional systems such as displays, warning/caution systems, and detailed engine simulations[1]. Along with this growth in functionality, the user community desiring access to the simulation grew. The simulations now supported flight deck research, multiple test benches, and the engineering community. Figure 1 shows growth of simulation size and name space.

The second area of growth is due to increased fidelity in the individual models. Simulations are now better termed "emulations" as most contain nearly the complete functionality of the operational systems. This level of simulation fidelity has many benefits in the life-cycle costs of airplane system engineering but it presents numerous challenges. To better handle the increased workload associated with these detailed models a graphical specification tool was developed. This tool, GSDS (Graphical Simulation Development System), provided users with an efficient means to

Copyright © 1997 by THE BOEING COMPANY.
Published by: The American Institute of Aeronautics and Astronautics, Inc.
with permission.

Figure 1. Simulation Growth



capture requirements while simultaneously creating specification drawings. The tool can automatically generate the simulation software from these drawings. Whether this tool helped users implement the additional fidelity requirements or its ease of use allowed users to create more is debatable, but either way we benefit from models that more closely represent actual systems[2].

A third area of growth was due to increased signal flow between systems. In test benches prior to the 777 program, 1000-2000 signals defined the interface between hardware and software. Another 2000 variables were used to pass data between the various system models. With the implementation of the ARINC 629 data-bus and all-digital-systems in the 777, the number of named signals grew to over 180,000. While all of these signals were not used in the simulation software, they were all potentially required for one or more test benches, and therefore needed to be included in the simulation.

The Emerging Problem

These three factors resulted in a 4-5 times growth in the size of 777 simulations from previous programs. Along with this growth in size came a substantial growth in complexity. It is difficult to put a quantitative number on complexity or its growth but complexity is known to grow faster than code size. Needless to say, fewer and fewer people were able to understand the details of the simulation, the data-flow between models, or the temporal dependencies of the different subsystems. Problems became harder to solve, coordination of changes took longer, and more people were required to perform the simulation software development process.

Our software development environment has also grown and changed over the years. However, one important factor has not changed. All of our avionics simulation models are written in FORTRAN. Whether written by hand or generated by GSDS, having a single language had always been seen as a means for easier integration and lower maintenance costs. However, the cross coupling of the simulation models caused by using FORTRAN commons to implement their interfaces produced an unmanageably high amount of system changes. Whenever any FORTRAN common block changed, all models referencing that common needed to be recompiled and re-released, even when that model did not change. During the early phases of the 777 program the interface name space was in a state of high flux, and therefore the commons derived from it changed every two to four months!

It became evident after several releases of the 777 simulation that our standard approach for dealing with

interfaces was not scaling up. We had decided to continue the practice of defining FORTRAN commons for interface variables. This is how we had always done it. We knew how to manage common based interfaces and we knew how to draw the configuration boundaries to minimize coupling. What we didn't recognize was the magnitude of the emerging name space or how its continuous change would impact our processes.

Our existing practice was to define interface variables for each model in a separately released configuration controlled area for each model. Other models that needed access to these variables would include the definitions from this area into their code. This works well as long as those interfaces are relatively stable. Whenever an interface is changed, every model that references that interface area must be recompiled and re-released. The dependencies are resolved by the compiler. The compiled objects (models) are tested and released as an integrated simulation. Users then build their specific simulation from the released object libraries.

The 777 program introduced an electronic ICD (Interface Control Drawing) to define the interfaces between the avionics systems on the airplane. At specific dates the interface definitions were frozen and distributed to suppliers and the ASL. Early releases of this database contained 30-40,000 definitions. This number quickly grew to 100,000 and now is over 180,000 entries used in the ASL. The database was released three to five times a year during the first years of the program, and has decreased frequency since then.

To process these ICD releases we developed a number of tools to parse and generate interface definitions and supporting software. This process was tedious and fraught with problems. Each model had to extract the definitions from the ICD for their system outputs. Commons were generated that defined these extracted ICD interface variables. Subroutines were generated to move data from the model's local variables to the ICD variables (these were called driven variables). Filters were used to group the variables into different commons based on whether they were actually driven by the current model. Subroutines were generated to pack and unpack bit fields contained in many of the ICD variables. Other subroutines were generated to provide gain/bias scaling on the output signals so that test benches could easily perform tests on signals. Often these generated files required hand modification to correct errors, thus further hampering the repeatability of the process. Figure 2 illustrates this process.

Numerous issues and problems were constantly being battled with this FORTRAN common based approach. The process to create the commons and supported code took days and sometimes weeks. The ICD contained errors that would not show up until the code was compiled. After fixing the problem in the ICD, the entire common creation process would have to be redone. Some of the generated code was so large that it would not compile. Once compiled, some of the routines took so long to execute that they were unusable. Additional filters were created to streamline these routines to the specific variables of interest. Many of these types of changes required rebuilding the simulation to tailor it for a specific test configuration (note minimum 40 minute time mentioned above). We were making it work but were expending a large amount of resources. Our standard approach did not scale up to the large number of variables in the ICD nor did it map well into our configuration control release process.

The difficulty was how to define an interface region that could be accessed by FORTRAN code, was not tightly coupled to the release process, and would not impact frame time performance requirements. The solution needed to provide bit field access and provide hooks for test bench signal testing. And of course, we should point out that we were in the middle of a fairly important airplane project and any schedule impact was unacceptable.

We had a problem! Unless we made some fundamental changes in the way we developed and managed simulation software, our effectiveness and efficiency would have gone to zero. Incremental improvements to our current software development environment would not result in the magnitude of improvement required. We needed a new approach, and that approach is "component software."

Simulation Component Software

Component software is a modularization that relies on predetermined constructs to eliminate dependencies and reduce the impact of those dependencies on the software development process. Classical FORTRAN software systems provide a good illustration. Common blocks provide for inter-module communication. The compiler generates object files containing variable name address offsets. The linker binds the object files, setting the absolute location of common blocks and resolving the offset addresses into machine addresses in the executable file. Changing a common block in any way, by adding or removing variables or by changing

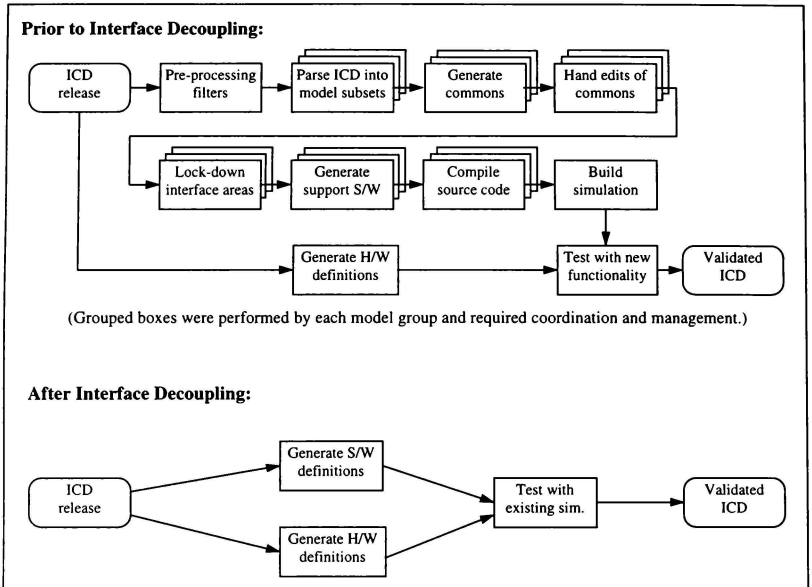


Figure 2 - Interface generation process repeated every ICD release.

variable types, and the referencing modules must be recompiled and the object files relinked into a new executable. For small software systems this edit-build-test cycle remains efficient and effective. Where the module source lines of code counts and the inter-model communication requirements are high - such as high fidelity air transport simulations - small changes to the interfaces cause large downstream effects. This ultimately effects the enterprise by increasing both cycle time and software labor costs.

Component software systems provide the framework where the impact of change is localized so that the effects of those changes on the overall system are minimized or eliminated entirely. Component frameworks provide mechanisms for:

- elimination of compile-time binding
- reduction of global data structures
- simplified interfaces between components
- as late as possible binding of the application

This approach allows independent development of individual components that are easily combined at a later time, resulting in reduced cycle time and reduced labor costs for the enterprise. See Table 1 for a summary characterization of component software.

	Component	**Monolithic**
Reconfigurability	High	Low
Link time	Low	High
Code Scalability	High	Low
Interface Scalability	Medium to High	Low

Table 1. Characterization of component software

ASL's First Cut at Components

The two most difficult requirements were the ability to postpone binding of the symbolic names and to have no impact on performance. It has been said that any computer science problem can be solved with an extra level of indirection. This was one that could not. Any additional indirection would be unacceptable. How do you move 15,000 variables around every 20 milliseconds and not impact timing?

The solution we developed was to postpone the final binding of symbolic interface variable names until simulation execution with the use of run-time code generation. At the startup of the simulation as each model first accesses its external data, a subroutine is generated. This subroutine performs the data movement between external global names and local model variables. Input and output subroutines are generated for each model and these are called before and after every execution of the model respectively. A derived requirement was that this solution not take more than 20 seconds additional startup time. This requirement and the performance requirement were both met. The 777 simulation performance was measured and showed virtually no change in frame time.

Some of the benefits realized with this solution which we called "Interface Decoupling" included elimination of numerous supporting routines, elimination of the hand-tailored simulations, and predictable execution time. See Figure 2 for an illustration of the improved process.

The big impact of this solution was on our processes. The 777 ICD was now treated as a separate entity and was not spread into the various model interface areas. The modelers no longer had the rigorous coordinated task of generating the commons and their associated source files. Processing the ICD was reduced to one or two weeks of one person's time. Figure 3 illustrates the savings in time and cost. The models are developed completely independent of the ICD process and only care about changes in the ICD when new model functionality is required. From the simulation's perspective, the ICD is like a large data-pool region. Since data location is determined at run-time, simulations can select which ICD to use without relinking the executables.

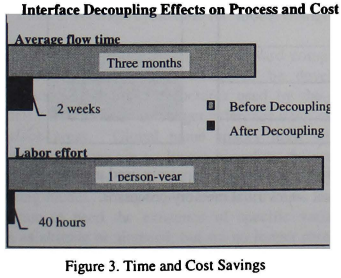


Figure 3. Time and Cost Savings

ASL's Component Framework

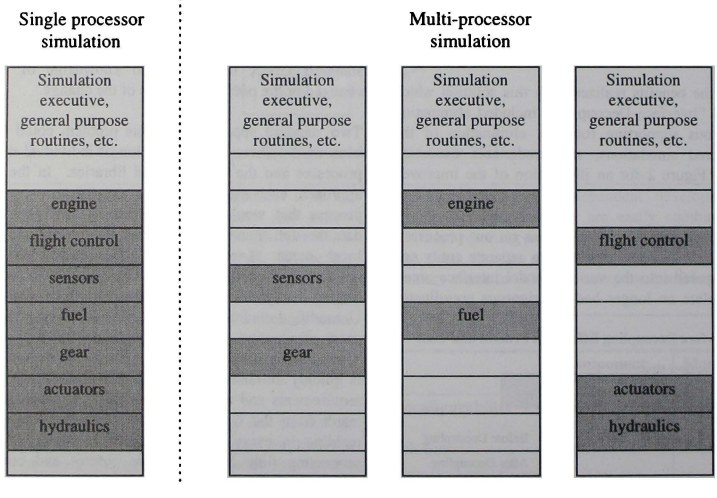
The next step towards component software in the ASL is to make components of the FORTRAN code. While Interface Decoupling provides some of the benefits of components, it still requires that we build large monolithic processes. This build step combines each of the model subroutines with ASL's simulation executive, general purpose math routines, and system support routines. To incorporate a model change, this time consuming step must be redone in its entirety. Even the smallest of changes requires this relinking step since the standard UNIX tools have no knowledge of what changed or the potential impacts of the change.

Two standard approaches to this problem could have been used. These are the creation of individual model processes and the use of shared libraries. In the first approach, each model would be created as a separate process that would run independently, access global data through shared memory, have its own independent local name space, and be synchronized by the simulation executive. Model changes would require linking only the model process that changed. Unmodified models could be shared and accessed by all users eliminating the large simulation build process. This solution would be ideal for numerous reasons but is quickly eliminated due to our stringent frame time requirements and memory limitations. The limitations result from the overhead of the simulation executive residing in every model process and the additional processing time for semaphore control and context switches of these processes. We are continuing to evaluate this approach for future implementations.

The second standard approach to our monolithic building problem is the use of shared libraries. Each model would be linked as a shared library component that can be loaded at run-time. The executable process

created at run-time is essentially the same as the monolithic process created by a static linker. The model libraries can be linked independently and changes can be isolated. Unmodified model libraries could be shared and accessed by all users. This process does not, however, eliminate the link step and has serious impacts to model performance. The linker still requires access to the shared model libraries to resolve global symbols and to reserve special regions used at run-time. While this does save time it was not as large a savings as anticipated. A bigger issue is the performance impact to the compiled code. The models must be compiled with special compiler options to generate position independent code. This creates data access inefficiency combined with additional instructions required to access external subroutines and results in models that run 35% to 45% slower than those statically linked. This performance penalty prevents us from meeting our frame time requirements and is unacceptable. This approach will be reevaluated if compute performance ever greatly exceeds real-time simulation compute requirements.

A unique solution to this problem has been prototyped and is being implemented in the ASL. It has features of each of the two typical solutions above and satisfies the performance requirements. The solution takes advantage of our simulation model hierarchy and brings back into use the old technique of overlays. This technique, for those that don't remember when 16K of memory was considered adequate, loads a subset of code into a predetermined offset in memory. Each individual model is converted into an overlay by statically linking to resolve all addresses. Additional directives are supplied to the linker to create the code and data at specific addresses and to satisfy global symbols. At run-time, each of the models are overlaid into their respective address regions and the short list of global symbols are resolved. Simulation memory layout is shown in Figure 4. The resulting executable process is functionally the same as the monolithic process created by a static linker.



Component models are loaded into each process at predetermined locations. The sparsely used memory regions in the multi-processor simulation have negligible impact on a virtual memory processor.

Figure 4 - Memory Map of Model Component Simulations

Features of this simulation architecture include:

- Models constructed as modular components
  - developed independently
  - interchangeable versions
  - independent name spaces
- Easy to share standard models
- Simulation build process virtually eliminated
- Delayed binding to interface variables
- Models not coupled to ICD or common structure
- Predictable run-time performance
- Run-time generation of buffer routines
- Users select components and execute simulation

The only additional step required is planning ahead for memory layout of the models. Each model is allocated a region of memory along with some room for growth. Models that grow past their allocated space are simply given a new region and the space is collapsed at the next major release.

The combination of these two methods, interface decoupling and model overlays, work together to provide nearly independent model development. The modelers can concentrate on their specific functional requirements and not be continually concerned about the cross coupling relationships between models. They can quickly implement changes and incorporate them into a full airplane simulation in minutes. See Table 2.

	Older Methods	**Component Methods**
Code change	20-40 minutes	3 minutes
Interface change	1 hr to 3-4 days	5-10 minutes

Table 2. Average cycle time for model changes

This solution was our first big step toward component software. It allows each model to be developed independent of the data structures used by the other models. The models no longer reference each other's interface areas. Global name spaces must still be maintained but we are no longer bound by the specific data structures or even the languages used by the defining model. Using this technique we know we will continue to need the existence of specific variables, such as altitude or airspeed, but we no longer care who owns them or what kind of data structure they are defined in.

Additional examples of component frameworks

Commercial off-the-shelf component frameworks exist on the market today. Although they are targeted for enterprise and PC desktop computing applications, applicability in the simulation software environment may not be far off. The two predominant standards are CORBA (Common Object Request Broker Architecture) and DCOM (Distributed Component Object Model.) CORBA is managed by the industry consortium OMG (Object Management Group) while DCOM is owned and managed by Microsoft and a growing list of industry partners. The problem space for these frameworks is similar to ASL's solution - predefined interface mechanisms that promote the independent development of components; as-late-as-possible binding of components.

Both frameworks decouple interfaces from component implementation(s) as well as provide for location transparency. As in the ASL approach to components, DCOM and CORBA both provide for physical separation of the component's interfaces. So once an interface is defined, there may be several implementations of the component that connects to that interface. Both CORBA and DCOM environments provide Interface Definition Languages as independent mechanisms for defining and implementing interfaces. In addition to a name, each interface is tagged with a unique identifier to avoid name conflicts in large software systems.

In addition to interface decoupling, these two frameworks provide for the transparent location of the components. In other words, using system registries and unique object identifiers, the client or container program need not know where a given component is located on the network. Calls to those programs within the client are as if the component program is on the host computer. Both CORBA and DCOM provide marshaling code that ensures proper data representation regardless of the component's host computer. See Table 3 for summary comparison.

Emerging Tools

In the desktop Windows 95 development environment, Rapid Application Development tools supporting the DCOM standard, such as Delphi and Visual Basic, are revolutionizing how software applications are being developed. With the exploding market of low cost, off the shelf components for an ever expanding range of uses, the make-buy decision is almost always "buy" and

the application developer's task evolves into one of software integration.

	CORBA	DCOM
1) Maturity of specification	In use for eight years	In use for one year. (COM for two years)
2) Operating systems supported	All mainstream non-Windows platforms	Windows and TBA: Solaris, HP/UX, MVS, Linux, Digital UNIX, others
3) Interfaces separated from component implementation	Yes	Yes
4) Component location transparency	Yes	Yes
5) Interface definitions	OMG IDL: language neutral. IDL compilers generate component language code	Microsoft IDL: user coded pointers and parameter passing mechanisms
6) Development software and tools	Purchase from third party suppliers	Part of Win32 SDK, Visual C++, Visual Basic 5.0
7) Interface definitions	OMG IDL: language neutral. IDL compilers generate component language code	Microsoft IDL: user coded pointers and parameter passing mechanisms

Table 3. Comparison of CORBA and DCOM

PC applications that used to take years are now being implemented in months. The user interface presented to the Windows developer remains constant - the component's methods, properties and events. Today's Windows software developer's toolset now includes a visual work space where components are dragged from the tool palate onto the work area and integrated by hand-coded "glue code" to ensure the proper performance of the integrated system. Custom code to perform a specific function is only developed when it cannot be purchased from a third party supplier. The standard interfaces defined by the underlying DCOM rules ensures that those components will connect to the larger application. The behind-the-interface details are unknown and irrelevant to the user. Only the embodied functionality matters.

One can imagine the simulation user interface for the airplane control system designer of the future. The palate displays icons for the airplanes systems. Specific versions of the systems are selected - Version A aero model, Version C propulsion , Version B hydraulic system, and so on. Press a button to build the simulation.

Further Study Required

Significant enterprise and desktop applications utilizing off the shelf CORBA and DCOM-based technology are successfully deployed today. Unanswered questions persist, however, with regard to the usefulness of these technologies in the air transport simulation domain. Will those technologies meet the hard real-time requirements of simulation frame times and data I/O? Only serious prototyping of simulations utilizing those technologies will tell.

Summary

Boeing's Airplane Systems Lab has developed and deployed a novel component architecture for the 777 air transport engineering simulation software. This architecture has enabled the reduction of FORTRAN software process development time by an order of magnitude without negatively affecting other processes. This has significantly reduced simulation software labor costs.

Other industry standard component architectures, such as DCOM and CORBA, have been developed for deployment in enterprise and desktop software domains, but not specifically for real-time simulation applications. Additional study is needed to determine their suitability to real-time simulations.

References

[1] Application of Graphical Software Generation Technology in Simulation of All-digital Transport Aircraft. Shane M. Fookes, et al, 1994 AIAA Modeling & Simulation Technologies Conference.

[2] Diagram Based Auto-generation of Simulation Source Code in The Boeing Airplane Systems Laboratory. K.C. Babb, 1996 AIAA Modeling & Simulation Technologies Conference.

[3] Object Magazine, October 1996. Choosing Between CORBA and DCOM, Mark Roy and Alan Ewald

THE FEASIBILITY OF AVIONICS SIMULATION VERSUS STIMULATION IN COMMERCIAL AIRCRAFT FLIGHT SIMULATORS

Keith D. Conzachi*
Flight Simulator Support Technician
American Airlines
Dallas, Texas

Abstract

The rapid development and complexity of avionics systems installed in the next generation aircraft has created a unique challenge to the simulation community, accompanied by millions of dollars in acquisition costs for airborne avionics components in commercial aircraft flight simulators. Simulator manufacturers assisted by avionics manufacturers are investigating alternative solutions to employing airworthy aircraft avionics systems for commercial flight simulation. This report will contrast issues on the benefits, risks, fidelity, reliability, initial costs, and life cycle costs involved with the decision to simulate or stimulate avionics systems for the next generation commercial aircraft flight decks.

Scope

Aircraft avionics systems have become far more advanced than most of us would have ever believed possible. Due to the sophistication of the next generation avionics architectures, avionics design and development issues have become more difficult and costly to the simulator user. Because of this, avionics and simulator manufacturers have been working on alternative solutions to simulating avionics systems for the next generation flight decks.

The decision to simulate or stimulate an avionics system is required early during the design specification phase of the flight simulator procurement. In the past, avionics system simulation was approached in several different ways by simulator engineers according to the technologies that were available during the development of each flight simulator. It often is an issue of pure cost, development time, and the availability of the avionics hardware for each aircraft being simulated. Recent advances in flight simulator avionics technologies have allotted us a few new alternatives to stimulating the avionics in our next generation commercial aircraft flight simulators.

To assess the feasibility of avionics simulation versus stimulation in the commercial aircraft simulator, the following areas of inquiry will be represented:

- Present demand for avionics simulation.
- Alternative avionics simulation applications.
- Fidelity issues.
- The decision to simulate or stimulate avionics systems.
- Reliability and maintainability issue.
- The ARINC 610 functionality challenge.
- Proprietary data issues.
- Updates as the aircraft configuration changes.
- The possible extinction of stimulation.

This report is geared towards the commercial airline simulator users that expect to procure next-generation aircraft simulators, or upgrade existing avionics systems within their flight training devices currently in use at their training centers.

Statement of Problem

Since the dawn of the integrated avionics systems found in the B757 and B767 flight decks, the flight simulation industry has been attempting to find a solution for the problems presented with integration of software driven avionics systems in flight simulators.

* Copyright © 1997 by Keith D. Conzachi. Published by the American Institute of Aeronautics and Astronautics, Inc. with permission.

Associate Member

Most integration problems frequently accompanied the fact that the manufacturers of the avionics systems were often very hesitant to provide data in order to accurately simulate the avionics systems, and almost as hesitant to provide the data required to successfully stimulate the aircraft avionics systems inherent to the flight simulator platform.

In an attempt at providing a solution to the avionics integration dilemma, the development of the ARINC 610 specification **"GUIDANCE FOR USE OF AVIONICS EQUIPMENT AND SOFTWARE IN SIMULATORS"** evolved. This specification provides the avionics manufacturer a standard set of functions that would permit the aircraft hardware to operate more efficiently in the simulated environment. More recent avionics integration efforts have dubbed this simsoft. Simsoft is a program internal to the avionics simulation program that enables an aircraft black box to communicate more efficiently in a simulation environment. Other operators have used a combination of adjusted training scenarios and modifications to the simulator host computer software in order to minimize the impact on the training mission.

Due to the rapid advances in computer technologies, alternative methods for avionics simulation have evolved, but stubbornness to accept the change in technologies has prevailed for most flight simulator users. The majority would rather select the stimulated approach over the risk involved in the simulated approach to integrating avionics systems in flight simulators.

Present Demand For Avionics Simulation

Several interviews with key industry personnel from the simulator, avionics, and airframe manufacturer's arena revealed a mutual interest in simulated avionics versus stimulated avionics systems simulation.

Avionics systems procurement costs escalate with every derivative aircraft being manufactured. The demand for avionics simulation from a direct cost-control standpoint will certainly increase in future flight simulator procurements. Avionics procurement and high life cycle costs due to the number of aircraft boxes and simulator hardware required is driving the simulator and avionics manufacturer to come up with alternative solutions to stimulating avionics systems in our next-generation aircraft flight decks.

Simulation of avionics systems can be very attractive to the simulator user at the airline and contract training center, it is not always the answer for every flight simulator user. Airframe manufacturers prefer stimulated avionics due to the developmental nature of the simulation task being performed. This gives them the ability to test and develop new aircraft avionics loads quite easily. Please note, from a manufacturer standpoint, that the airborne avionics boxes are readily available from the avionics manufacturer due to the aircraft certification processes taking place at these locations. Airframe manufacturers are also the first to receive updates for avionics systems as they become available from the manufacturer. Unlike the airframe manufacturer, most simulator users do not benefit from this luxury.

Stimulation therefore makes sense from an engineering and integration environment, but I believe the airlines will demand alternate applications like the rehost or retarget solution due to cost control measures alone. Simulator users I have spoken with believe that simulation of avionics systems utilizing the rehost or the retarget method is the future and will become the accepted standard in a few more years. The more I research the situation, I tend to agree with the simulator industry.

Alternative Avionics Simulation Applications Available

In recent year's simulator manufacturers have been driven to reduce the procurement cost of flight training simulators. The largest recurring cost in the simulator procurement has been employing actual airborne avionics hardware. One by one, the avionics hardware has been replaced with simulated counterparts that are functionally equivalent to the authentic units, but implemented with a lower cost commercially available hardware. The goal of the alternate simulation applications is to automate the once manual process of simulating and stimulating each specific avionics system being designed. In the light of alternate simulation techniques, two more recent developments for simulation of avionics systems have evolved. Both applications achieve the same common goal of replicating the functionality of the actual aircraft unit. They are the retargeting and the rehosting of avionics hardware by recompiling the avionics operational software program on to a commercial computer platform.

Retarget. With this simulation application, you will recompile the actual avionics system vendor source code to the target software language of the host computer platform. With this approach, the retargeted software is treated like any other program on the host computer. In some cases the simulator host computer would not be capable of supporting these demands for spare CPU time rendering the solution of more complex avionics applications to the rehost application or any remaining simulation substitute.

The main advantage of this approach is that the aircraft hardware is not required for the simulation. In the case of the B777 AIMS, this is a very significant cost saving The next generation flight management systems primarily make use of the ADA software language. With the retarget method the avionics vendor source code would be downloaded to the host computer and by utilizing an automated software translation tool transform the operational program to C language capable of running on the host computer platform. Some of the more unfavorable points might be due to utilization of avionics source code versus executable code; the fidelity of the avionics load could be questionable versus the actual aircraft hardware. This occurs during compilation of the avionics vendor source code. Honeywell utilizes specialized compilers that match their design methodology. Getting equivalent functionality will require manual intervention and modifications to standard compilers used in simulator host computers.

Rehost. This application in many ways is the same as the above. To rehost, the avionics vendor executable code is taken and translated to the appropriate host software language and installed on a dedicated host computer platform. The host computer is used solely for the avionics application being simulated and interfaced to the simulator host computer via specified communication protocol in order to communicate with other aircraft systems within the flight simulator platform.

With this approach, the avionics hardware is not required representing an initial cost saving as well. Keep in mind, if the rehost computer software language is the same as the avionics software code the implementation is simplified and can be installed directly on to the rehosted computer platform.

Advances in simulator technologies have led to fully integrated simulated solutions for the FCC, EICAS, GPWS, TCAS, and WXR avionics systems eliminating the requirements for the actual aircraft hardware. Applications presently available for the rehost and retarget applications according to research material acquired are the AIMS, DEU's, FMS, and EICAS avionics systems. Continued research could open up further possibilities I am certain although these alternatives have been developed and tested, each will have their positive and negative points inherent with the simulation of actual avionics hardware.

Fidelity Issues

Within the simulator community, the simulated approach of avionics systems had often been assumed to be of lower quality, and the fidelity of the simulated model did not function exactly like the actual aircraft hardware. Airline pilots are also a bit skeptical that the simulated avionics systems can give you the throughput and precision required to function like the actual aircraft hardware. With the aircraft hardware approach, the operator is assured that the avionics unit will have the exact functionality of the avionics box being simulated. Following discussions with several flight simulator maintenance and engineering personnel utilizing simulated, stimulated, and alternate avionics simulation applications the following fidelity issues were noted.

With previous simulated and stimulated avionics applications, it was a straight forward answer for most simulator users: stimulate the avionics systems. With he stimulated approach, you could not go wrong, most replied. With simulator avionics interface hardware and the appropriate avionics integration technique, simulator manufacturers were capable of completing an avionics integration effort. With this method you were assured the simulation would be as functional as the actual aircraft avionics hardware. But the aircraft avionics box would only offer minor allowances for simulation use. Typical simulator systems will endure situations that in most cases would never be seen nor even attempted in the actual aircraft. Simulators in most cases operate seven days a week, twenty-four hours a day. In one year most simulators will see more takeoffs, landings, engine starts, engine shutdowns, and instructor induced malfunctions than the aircraft would ever see during its entire life-cycle. Flight simulators have been referred to by avionics manufacturers as the avionics systems test bed for these very reasons. If it will operate in the simulated environment, it is sure to function correctly in the aircraft.

In fact most avionics changes are first tested in the simulator to observe developmental avionics software loads.

Consequent to these consideration's one might want to reconsider the simulated approach rather than a stimulated approach to avionics integration in flight simulators.

The main advantage of the simulated approach was the ability to modify the avionics program to integrate into the simulated environment more freely Also simulation would minimize the use of simulator interface hardware required to integrate the avionics simulation although the negative impact of simulating avionics systems is the avionics vendor data issue. To date, this is one of the most difficult factors that needs to be overcome by the flight simulation industry. The ability to accurately simulate the specific avionics components lies in the hands of the avionics vendor to provide accurate data in a timely manner, rendering a functional avionics simulation design.

Simulator users of the retarget and the rehost applications have had very favorable results concerning simulation fidelity, reliability and maintainability issues. Indications from applications presently used in training with the retargeted and rehosted avionics seem to be adequate substitutes for the actual aircraft avionics hardware. With the retarget and rehost approach to avionics simulation we are assured that we are getting an exact simulation of the avionics component being simulated. These applications are more or less utilizing the avionics operational source or executable code, which results in a higher fidelity simulation and aircraft functionality. It takes away all the guessing we once endured with the earlier applications for simulating avionics systems or stimulating them, for that matter.

In conclusion, fidelity is primarily based on the quality of data acquired and the quality of the avionics simulation design being implemented.

The Decision to Simulate or Stimulate Avionics Systems

Special circumstances warrant consideration of a stimulated avionics solution over a simulated avionics solution for flight simulation. Where production volume is expected to be low, a stimulated solution can be most effective particularly for a mature aircraft system like the B737, B757, B767 or a Canadair RJ, where a simulated solution has already been designed and outfitted for these aircraft systems. A stimulated solution may also be more feasible if the customer is able to provide its own avionics hardware in the case of a commercial airline or an airframe manufacturer. In the case of the contract training centers like FlightSafety or Simuflite, the stimulated solution may be more economical than the simulated solution. However aircraft like the B777, MD11, or the A320 are more feasibly perceived with a simulated solution rather than a stimulated one due to the complexity and high dollar figure for the avionics systems installed.

With the hardware stimulated approach, there will be higher procurement costs for the aircraft avionics hardware. There will also be higher life cycle costs due to the number of LRU's and simulator interface hardware required to support the avionics simulation. However there is typically less integration time involved with the stimulated approach,

The software simulation will require lower procurement costs due to minimal hardware interface requirements, but software development cost will generally be higher. The benefits from the simulated approach are that avionics hardware is not required, which means fewer avionics spares are required representing lower avionics acquisition costs for simulators and that updates are limited to man hours, because update disks are supplied with each aircraft. Simply stated, the life-cycle cost of having simulated versus stimulated is no-contest. I am sure the avionics manufacturers and simulator end-users would agree; the simulated approach is the more cost effective measure.

In conclusion we utilize simulated avionics versus stimulated avionics because they are less expensive, require less hardware, are easier to maintain and update, and have proven to be a more flexible system in avionics simulation for flight simulators.

Reliability and Maintainability Issues

With the stimulated approach, you will require significant amounts of aircraft and simulator interface hardware, which translates to lower reliability and increased downtime when compared to the simulated approach. Downtime may also result from LRU fault isolation, although diagnostics can be performed utilizing built in test equipment "BITE" inherent to each aircraft box. The negative impact of this situation is that you cannot troubleshoot during

training. Aircraft spares are also required which will add additional expense with the stimulated situation. With the simulated approach, minimal avionics and simulator interface hardware is required. This translates to higher reliability and reduces simulator downtime. Minimal spares are necessary for the simulated approach, unlike its stimulated counterpart.

Please note, at the time of research there was insufficient data compiled for the reliability and maintainability issues for the retarget and the rehost applications for avionics systems simulation. This will be provided at a later date when the data is available.

Commercial airlines typically have the luxury to draw from a stock of avionics at each respected jet base, so maintainability from a stimulated standpoint is usually not a problem though the black boxes are only as good as their stimulated inputs and ARINC 610 functionality permits them to be. Often we experience anomalies in the simulator that the aircraft will not experience; with this in mind, simulated solutions are a more favorable resolution to handle these specific protocols.

The ARINC 610 Functionality Challenge

The simulator industry is attempting to use software modeling of avionics, some systems may still require the use of the actual aircraft hardware. This occurs due to system complexity, proprietary data, and unavailability of data. From all the simulator users interviewed, ARINC 610 is necessary for future avionics simulation. Strong efforts from the airline and the simulator manufacturers to adopt this standard from aircraft conception is in place, but the adoption of this standard in initial design for both airframe and avionics manufacturer is a very slow and frustrating issue.

Integrated avionics systems for the B777, MD11, A320 and the B747 have increased the use of ARINC 700-series avionics sensor devices to provide a highly automated cockpit. Integration of these types of equipment brought a whole new range of complexity in avionics integration in flight simulators. In the 1990's a new generation of avionics was conceived as Integrated Modular Avionics (IMA). This concept compromises complex functions that are packaged together to share common resources. It is inevitable that these will replace the ARINC series 700 box, posing yet another challenge for avionics simulation efforts.

With next-generation aircraft like the B777 and B737, the ARINC 610 specification was designed and approved subsequent to the design definition for the B777, so it was not incorporated into this airplane. For the B737, a revision of the FMS is being used from previous B737 models, so ARINC 610 will not be a functional part of that aircraft system either. Boeing has suggested that ARINC 610 guidance be incorporated into future avionics technology. Beyond this, the industry should be considering follow up guidance addressing a high level of function integration and redundancy introduced in integrated modular avionics.

Avionics manufacturers are aware of ARINC 610 and have not turned their backs on it. This is no small task and requires a substantial work effort to correlate avionics systems design with ARINC 610 specifications for present and next-generation flight decks; repeats an avionics representative. They are looking for a way to share the cost with the industry because of the unique endeavor proposed. The cost and time factors required to tackle a project this size is enormous and typically they feel this should not be the sole responsibility of the avionics manufacturers. After all, avionics manufacturers are in the business to design and build avionics systems for air transport, business aviation and general aviation aircraft. From an avionics manufacturing standpoint, the ratio of simulators to aircraft can not compare. But to the airline training center or any other training center for that matter, lacking a viable solution to ongoing problems experienced with integrating avionics in simulators could mean delays for functional aircrew training devices in the future.

The simulator manufacturers have said it is not necessarily true that there has to be ARINC 610 functions designed into the box. There are other software solutions for manipulating the box to achieve the same ARINC 610 functionality. As far as the rehost and retarget approach, they are quite capable of handling the ARINC 610 functions. although each piece of equipment and each manufacturer needs to be handled on a case by case basis. It's not a situation that can guarantee that a rehost solution is always going to be a feasible solution, but it can be an alternative path to relieve some of the burden experienced the avionics manufacturer.

In conclusion, even though effort has been put forth to include these specifications in future avionics hardware design specifications, there is a long road ahead before we see it fully implemented in future avionics system design.

Proprietary Data Issues and Requirement for Avionics Simulation Integration

After discussion with key industry personnel, there were mixed feelings between most; data issues continue to be an ongoing battle with avionics integration in flight simulators. Past experiences have prevailed in either case of the simulated or the stimulated approach. Data is the key to the fidelity and functionality of the aircraft box being simulated. In the past there have been problems acquiring accurate data for initial system design and even more difficulty with acquiring data for updates to aircraft systems. What this translates to is a major slowdown in avionics integration efforts. It is more difficult with aircraft like the B777 and the MD11, two of the more recently certified airframes. This is due to the development nature of the aircraft being simulated. Data is changing rapidly and data received for integration efforts is questionable at most. Usually due to data issues, simulator manufacturers tend to stimulate avionics systems until the aircraft stabilizes to the point where they can attempt a viable simulated solution for the aircraft avionics hardware.

In the case of the retarget approach, the source code is obtained from Honeywell and recompiled on the host computer platform. Data required for future updates is allegedly covered under the master data agreement purchased with the simulator.

In the case of the rehost, this approach allows you to utilize the operational program diskettes used to update the aircraft black boxes and load them directly on to the rehost platform. Again the master data agreement allows the airline to download the software upgrades as they appear.

Recent discussions with simulator users of the B777 have recognized the avionics manufacturers more open to assisting with avionics integration problems experienced within the flight simulator community in respect to the rehost and retarget approach to avionics simulation. They appear to want to become more involved with future avionics integration efforts and are assisting the simulator manufacturers with future avionics system integration requirements. Unfortunately, I believe the same holds true for issues discussed earlier concerning stimulated and simulated applications.

Updates to Avionics Systems as the Aircraft Configuration Changes

Simulated systems have been proven to be easier to update as aircraft operational changes take place. This is more of an issue with the developmental or pre-certification aircraft like the B777, and the MD11. Aircraft with mature avionics systems like the B757 and the B767 are stabilized and very minor changes at most will occur with these avionics systems. In some cases the avionics manufacturer may even replace the entire aircraft box. The only significant problem would arise during a major hardware change in a specific avionics system. Hardware design changes of this complexity could require a significant amount of hardware re-design or software development work to update the simulated or stimulated avionics system. We also have suffered the wrath of receiving accurate data from the airframe or the avionics manufacturer to accurately implement the changes that have occurred.

The stimulated version is much easier though. Usually it will require a simple software upgrade to the aircraft avionics box accomplished by loading a new operational program into the aircraft box. When time and certification are important, the stimulated version in this instance would be much easier. Data is usually slower coming with a post-aircraft certification than with the pre-aircraft certification.

Please note, despite the very negative response with respect to simulating avionics systems, there are difficulties that arise in the stimulation of avionics systems as well. For instance, due to the developmental nature of the B777, we often experience difficulties acquiring avionics components along with proper documentation needed to accurately design each specific system being stimulated. Updates as well are not a sure bet. Depending on the magnitude of changes, you may not be ensured you a plug and play situation. Thus, there are difficulties with both applications.

Overall, updating stimulated avionics still appears to offer the simple approach to avionics upgrades as the aircraft systems change. A normal maintenance and engineering effort is still the case with most stimulated avionics changes, but due to the initial phases of the alternative simulated solutions, time will tell if they offer the same level of simplicity offered by its stimulated counterpart.

In conclusion, the obvious fact that concerns the airline industry is the ability to update the simulator systems as the aircraft avionics configurations change, and the ability to support the simulator through out its life-cycle following certification with out the direct support of the flight simulator manufacturer.

Will Stimulation Become Extinct?

Customer preference will drive the avionics simulation versus stimulation decision in present or future flight simulator procurements. There are many systems in the aircraft that have a more feasibly stimulated solution than a simulated solution due to the relatively low cost and minimal complexity of the specific avionics system. Simulation is targeted more towards high cost and complex avionics system

Conclusion

Clearly, action is required to curve the high cost of buyer furnished equipment installed in present day commercial flight simulators. As costs escalate for advanced avionics platforms in the next-generation aircraft like the Boeing 777, new technologies are required for alternative avionics simulation integration and future updates during the life cycle of today's commercial aircraft flight simulators. By addressing the areas we have discussed in this paper, it is obvious that technology has made both approaches a feasible solution to simulating avionics in flight simulators in both the present and future. Each approach has its pros and cons. It will take more time and research to evaluate new technologies that have emerged to determine what will be the more feasible solution for avionics simulation. Overall, customer preference will drive the simulation or stimulation issue for avionics systems simulation in flight simulators.

Acknowledgments

I would like to personally thank the following organizations and individuals for this input and research material required to make this report a reality. They are as follows:

- Bob Glenn, United Airlines, Denver
- Keith Chabot, Delta Airlines, Atlanta
- ARINC
- Flight Safety International
- Honeywell Avionics
- Rockwell/Collins Avionics
- CAE Electronics
- Thomson Training and Simulation
- Boeing Commercial Airplane
- American Airlines Engineering Staff

References

- FSEMC Publication 95-294/FSG-012 Sept 1, 1995. FSEMC Report
- FSEMC Publication 96-256/FSG-23 November 15, 1996. FSEMC Report
- ARINC Report 610A Guidance For Use Of Avionics Equipment And Software In Simulators. February 1, 1994
- Simulation of Electronic Flight Instruments in Ground Training Devices. Michael R. Skerrett, Hughes Simulations Systems Inc.
- Simulation of Aircraft System Controllers: A Discussion on the Feasibility of Automatic Code Generation. Mr. R. Boisse, CAE Electronics Ltd. Canada

AIAA-97-3797

SIMGRAPH - A FLIGHT SIMULATION DATA VISUALIZATION WORKSTATION

Joseph A. Kaplan
NASA Langley Research Center
Hampton, Virginia

Patrick S. Kenney
UNISYS Corporation
Hampton, Virginia

Abstract

Today's modern flight simulation research produces vast amounts of time sensitive data, making a qualitative analysis of the data difficult while it remains in a numerical representation. Therefore, a method of merging related data together and presenting it to the user in a more comprehensible format is necessary. Simulation Graphics (SimGraph) is an object-oriented data visualization software package that presents simulation data in animated graphical displays for easy interpretation. Data produced from a flight simulation is presented by SimGraph in several different formats, including: 3-Dimensional Views, Cockpit Control Views, Heads-Up Displays, Strip Charts, and Status Indicators. SimGraph can accommodate the addition of new graphical displays to allow the software to be customized to each user's particular environment. A new display can be developed and added to SimGraph without having to design a new application, allowing the graphics programmer to focus on the development of the graphical display. The SimGraph framework can be reused for a wide variety of visualization tasks. Although it was created for the flight simulation facilities at NASA Langley Research Center, SimGraph can be reconfigured to almost any data visualization environment. This paper describes the capabilities and operations of SimGraph.

Introduction

SimGraph was designed to improve comprehension of the large amount of data produced from flight simulation tests in the Flight Simulation Facility at NASA Langley Research Center (LaRC). Data produced from the flight simulation is presented by SimGraph in several different formats; these include 3-Dimensional Views, Cockpit Control Views, Heads-Up Displays, Strip Charts, and Status Indicators. Since these displays may not satisfy all of the needs of the research community, SimGraph can be easily modified to accommodate the addition of new graphical displays into its existing framework with minimal changes, thereby customizing the software to the users' particular environment. These graphical displays can be taken from existing programs and easily integrated into SimGraph.

High Level Design

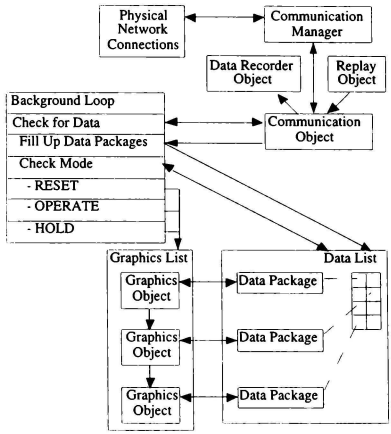


Figure 1 - High Level Design

The foundation of SimGraph is a C++ object-oriented framework. This object-oriented framework handles the communication and data transfer that is standard with all of the NASA LaRC's Flight Simulation Facility data visualization software. The framework has been thoroughly tested and is continually re-used, thus saving valuable programming resources. With each use, the framework is tested again, improving the quality of the software. The graphical displays, which sometimes are unique to each simulation, are then inserted into this

framework. As new displays are developed, they are added to an existing library. This library is an additional source of individual software components that can either be re-used or slightly modified. For example, a graphical display showing the cockpit controls of one fighter aircraft can be easily modified to show the cockpit controls of another fighter aircraft.

Object-oriented design allows a system to be viewed as a collection of objects and the interaction between those objects. This interaction occurs between the objects' interfaces (Figure 2). The implementation details of the objects are encapsulated behind the interface. Since the rest of the program cannot access the implementation details, all interaction with the object must take place via the interface. This forces clearly defined interfaces and aids in designing modular software. If all connections to an object are through the interface, then the implementation details of an object may be changed without affecting the rest of the system as long as the interface does not change. This simplifies maintenance of the software since errors within the object can be corrected without causing errors elsewhere in the program. The entire object may also be removed and replaced with another object that has the same interface.

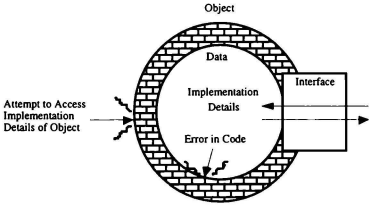


Figure 2 - Object-Oriented Design

SimGraph was designed to be computer platform independent in several aspects. It is written entirely in C++. SimGraph utilizes the X Window System to manage its graphical displays. The X Window System runs on a wide variety of computer platforms, ranging from supercomputers to personal computers. Anything that can be drawn inside an X Window can be used to build a graphical display. Several of the displays in SimGraph were written in the OpenGL graphics description language. The manner of communication between the simulation program and SimGraph is also platform independent. SimGraph can exchange data with the simulation program using Internet Domain Sockets, UNIX Domain Sockets, SCRAMNet Replicated Shared Memory from Systran Corporation, or the Advanced Real-Time Simulation System (ARTSS)[1], which is the NASA LaRC's real-time simulation input/output system. New methods of communication can be easily integrated into the program by extending the communication interface, which is encapsulated within a single object.

Detailed Design of SimGraph

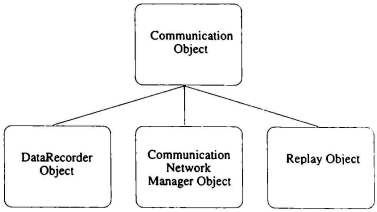


Figure 3 - Overview of the Communication Object

The Communication Object (Figure 3) is SimGraph's link to simulation data. All outside communication, either with a simulation program or a data file, is handled by the Communication Object. This includes both receiving and sending data. During a simulation run, the user may choose to save the data in a file so the data can be reviewed at a later date. When reviewing the data, the Communication Object obtains the simulation data from a file instead of a connection to the simulation program.

The Communication Network Manager Object is SimGraph's link to the simulation program that supplies it with data. The user is allowed to select from several methods of communication. The SimGraph program uses the same function calls to the Communication Object regardless of the method of communication selected by the user. This is possible because all of the implementation details of the communication have been encapsulated behind the interface. Adding new methods of communication is as simple as editing the Communication Object to handle those methods and recompiling the program. The rest of the SimGraph program will not be affected by those changes.

The Data Recorder Object archives the data transferred between the Communication Object and the Communication Network Manager Object. The Data Recorder Object is created when the user requests data to be saved, recording data from that time forward. SimGraph does not store any unnecessary or old data because of the

amount of memory that would be necessary to accommodate it.

The Replay Object of SimGraph allows the user to examine data from a previous SimGraph session. The data may be viewed at different speeds and directions. All the information for the displays will be loaded from a data file instead of being communicated from the simulation program.

Data Storage

The next portion of the SimGraph framework is the Data List (Figure 4). The Data List manages the data used by the Graphic Objects that are currently being viewed by the user. Because of the policy used to manage the data, a particular packet of data will only be transmitted once, regardless of the number of Graphic Objects that need it. A Graphic Object can be closed and the data that corresponds to it will continue to be transmitted unless it was only in use by that particular Object.

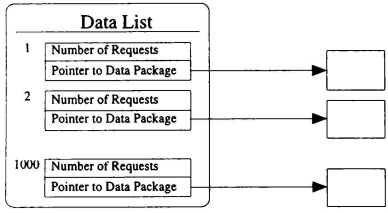


Figure 4 - Diagram of the Data List

The most important function of the Data List is data parsing (Figure 5). Data is sent from the simulation program in a large block. This block is the actual binary data which has been copied into a character array for transmission. When the Data Block is received by SimGraph, it is immediately passed to the Data List, which will parse the data, ensure that it is correct and has not been corrupted during transit, and break the data into the individual components for retrieval by the Graphic Objects.

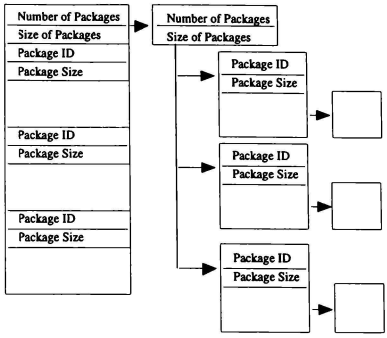


Figure 5 - Parsing of the Data Block

Graphic Objects

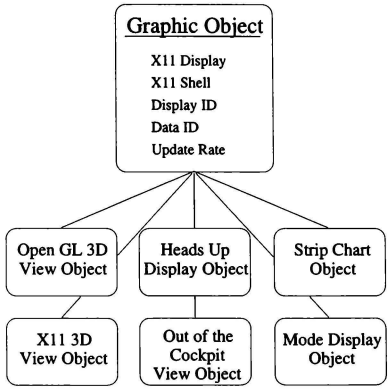


Figure 6 - Overview of Graphic Object

The Graphic Object is the cornerstone of SimGraph (Figure 6). Each display that is shown on the screen is represented by a Graphic Object. All of the Graphic Objects in SimGraph are derived from a generic parent object. This parent object is merely a skeleton of all the attributes that SimGraph Graphic Objects have in common.

The Graphic Objects have three methods in common which correspond to the three simulation states:

RESET, HOLD, and OPERATE. The initialize method is called when the Graphic Object is first created and during RESET. This routine will initialize all variables. The initialize method must be set up so that it can be called multiple times without causing any type of error (i.e., allocating without de-allocating memory, creating Widgets without destroying them, etc.). The second method is the update method, which is called when the simulation is in OPERATE. The Graphic Object's Data Package is sent to the Graphic Object, along with the size, when this method is called. The Graphic Object will unpack its data and make its graphic update appropriately. The third method is the hold method, which is called when the simulation is in HOLD.

Sample Graphic Objects

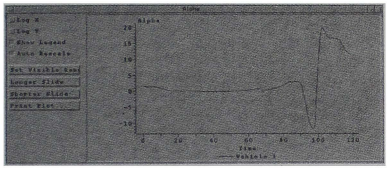


Figure 7 - Strip Chart Graphic Object

The Strip Chart Graphic Object (Figure 7) was designed to plot data sent from the simulation program. This Graphic Object will plot data on its drawing area, updating the plot whenever new data arrives. The Strip Chart Object is also capable of displaying multiple plots of data on the same drawing area. Each plot can be drawn with a distinctive line pattern to differentiate the separate plots. The main widget used for this Strip Chart Graphic Object was provided by Fermi National Laboratory[2].

The Mode Indicator Graphic Object (Figure 8) presents the user with the current time of the simulation run, the mode of the simulation, and the number of vehicles currently in the simulation. The mode the simulation is in is indicated by a red box. In Figure 8, the simulation is in Operate.

The OpenGL 3D View Graphic Object (Figure 9) draws the simulation aircraft in their correct orientation and allows the user to move the viewpoint around these aircraft. By moving the viewpoint, the user can gain new insight into what is actually occurring with the simulation aircraft. The user can also easily understand the physical orientation of the vehicles and relative positions to one another. The OpenGL graphics system is a software interface to graphics hardware that runs on multiple platforms[3]. The software for this display was originally taken from the VISION software package[4]. User configurable ground traces, altitude traces, aircraft snapshots, and multiple viewpoints are just some of the new features that have been added.

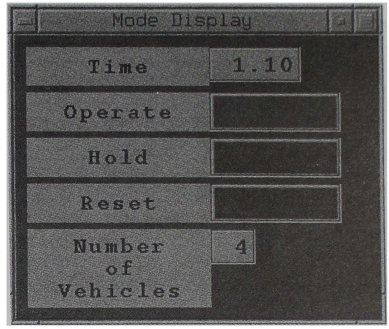


Figure 8 - Mode Indicator Graphic Object



Figure 9 - OpenGL 3D View Graphic Object

The X Window System 3D View Graphic Object (Figure 10) was designed as an alternative for the OpenGL 3D View Graphic Object. The main difference between the two Graphic Objects is the 3D View

Graphic Object utilizes the X Window System for its drawing routines as opposed to the OpenGL Graphics Language. While the X Window System does not have the speed or the complexity of OpenGL, it does run on a greater number of platforms[5]. The LibV Graphics Library used to draw this object was taken from ACM, a multi-player flight application developed by Riley Rainey[6].

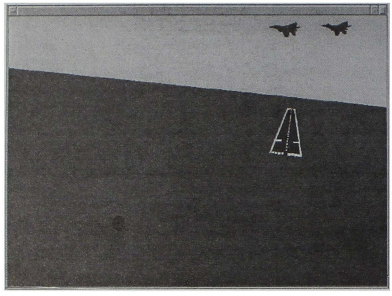


Figure 10 - X Window System 3D View Graphic Object

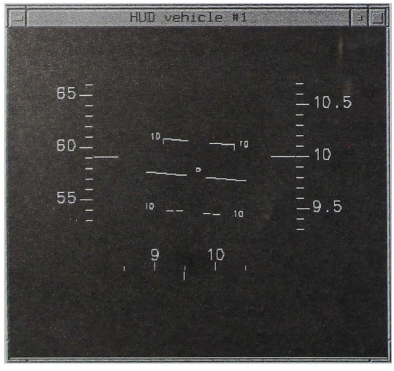


Figure 11 - Heads-Up Display Graphic Object

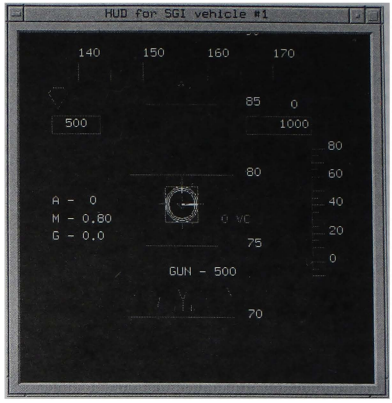


Figure 12 - OpenGL Heads-up Display Graphic Object

The Heads-Up Display (HUD) Graphic Object (Figures 11 and 12) displays a representative view of the pilot's HUD. HUDs typically give information about the current state of the vehicle, such as heading, pitch, roll, altitude, and velocity. Additional information, such as weapon status, fuel state, and rates of closure to other aircraft, is sometimes included and can easily be added to the HUD Graphic Object. The object shown in Figure 11 utilizes the LibV Graphics Library to draw its information. The object in Figure 12 uses the OpenGL graphics system.

The Out-of-the-Cockpit View (OCV) Graphic Object (Figure 13) gives the user a representative view of what the pilot is seeing. The OCV Graphic Object plots the locations of the simulated vehicles and positions the eyepoint at one of these locations. The resulting view is then shown. Two separate OCV Graphic Objects are available. The Graphic Object shown in Figure 13 is the OCV-HUD Graphic Object. This object has this name because a Heads-Up Display is superimposed upon the OCV View. An additional OCV Graphic Object is available without the HUD and is referred to as the OCV Graphic Object. This object also utilizes the LibV Graphics Library.

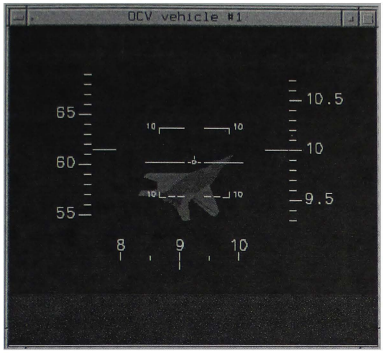


Figure 13 - Out-of-the-Cockpit View Graphic Object

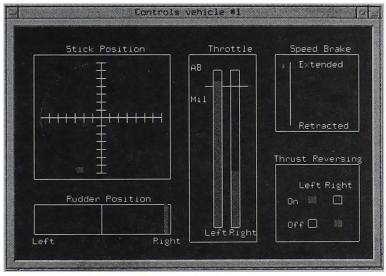


Figure 14 - OpenGL Cockpit Controls Graphic Object

The Cockpit Controls Graphic Object (Figure 14) displays a current view of the cockpit controls. The stick position display has a ten second trace. Some of the information displayed includes stick, rudder, throttle, and speed brake positions.

The OpenGL Aspect Circle Graphic Object (Figure 15) shows the three dimensional perspective view of the other aircraft involved in the simulation in relation to the current aircraft. One circle shows the location of the other aircrafts in vertical relation to the current aircraft (i.e., are they above or below the current aircraft). The other circle shows the location of the other aircraft in horizontal relation to the current aircraft (i.e., are they to the left or the right of the current aircraft). Both circles give information regarding whether the other aircraft are behind or in front of the current aircraft. In Figure 15, there are two aircraft at the same altitude as the present aircraft. One is on the right side of the present aircraft and the other is on the left side.

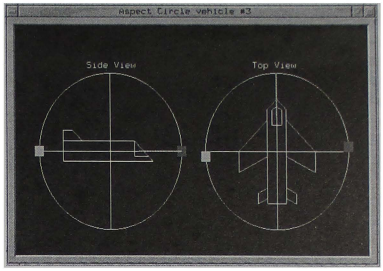


Figure 15 - OpenGL Aspect Circle Graphic Object

Graphic List

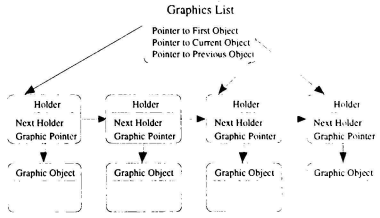


Figure 16 - Overview of Graphics List

The last portion of the SimGraph framework is the Graphics List (Figure 16). The Graphics List is a linked list that is composed of Graphic Objects. Graphic Objects are added to this list when the user requests a new graphic to be displayed on the screen.

Data Flow

Figure 17 shows the high level data flow inside of SimGraph. When the user requests for a new Graphic Object to be displayed on the screen in step 1, the interface passes the message to the Communication Object. The Communication Object transmits the request to the real time program in step 2. The real-time simulation program packs the data for the new Graphic Object along with the data for all of the previously requested data. In step 4, a new Data Block

is sent to the SimGraph Computer. The Communication Object reads in this Data Block in step 5. The Data Block is then passed to the Data List to be parsed and split into individual Data Packages in step 6. Step 7 finishes the data flow when the Graphic List updates each of the graphics by passing the requested data.

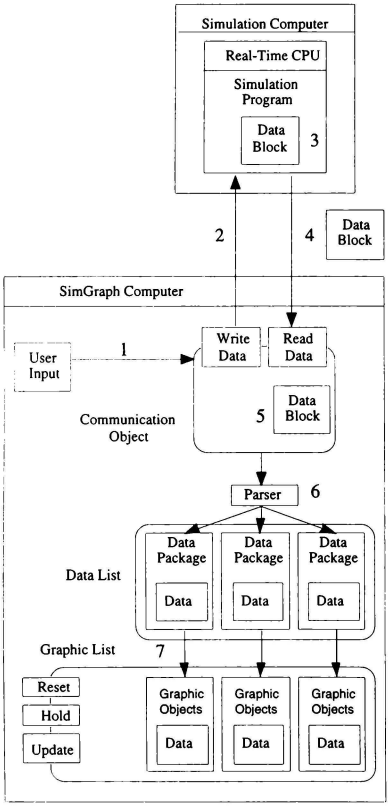


Figure 17 - High Level Data Flow

User Interface

The first window that is presented to the user is the Communication Selection Window (Figure 18). This window allows the method of communication between SimGraph and the simulation program to be selected.

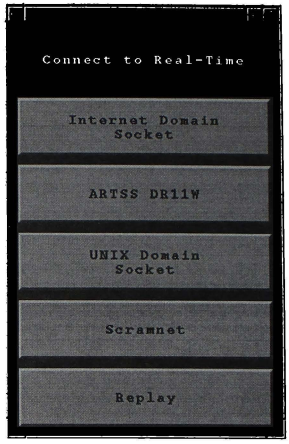


Figure 18 - Communication Selection Window

If any option other than Replay is selected from the Communication Selection Window, the Main Option Window in Figure 19 will be shown. This window allows the user to save data, open Graphic Objects, close Graphic Objects, load (or save) User Configurations, and quit the program. This window stays on the screen for the duration of the SimGraph program.

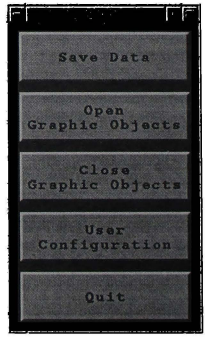


Figure 19 - Main Option Window

The Save Data Button in Figure 19 allows access to the Save Data Window (Figure 20). This window enables the user to start or terminate saving data to a file.

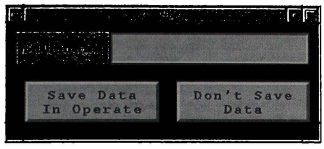


Figure 20 - Save Data Window

The Open Graphic Objects Button in Figure 19, when selected, presents the user with the Open Graphic Objects Window (Figure 21). The Open Graphic Objects Window is the only method for the user to select Graphic Objects to be presented. This window will change depending upon which Graphic Objects are available.

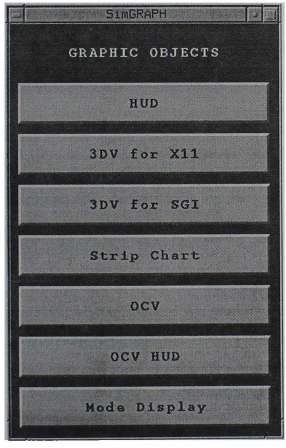


Figure 21 - Open Graphic Objects Window

Different actions can occur based upon the Graphic Object which is selected from the Open Graphic Objects Window. If the Graphic Object selected depends upon the number of vehicles in the simulation, a window similar to the HUD Selection Window (Figure 22) is presented to the user. In the case of Figure 22, the simulation has two vehicles in it. If there were four vehicles in the simulation, the HUD Selection Window would present the user with four buttons, HUDs for vehicles one through four. If the Graphic Object does not show an individual vehicle status but instead presents information about the entire simulation, such as the Mode Display, then pressing the button on the Open Graphic Objects Window would cause that Graphic Object to be created. Selecting the Strip Chart Button causes SimGraph to present the user with the Strip Chart Selection Window (Figure 23). The window that the user selects from (i.e., the HUD Selection Window or the Open Graphic Objects Window) disappears after a button is pressed.

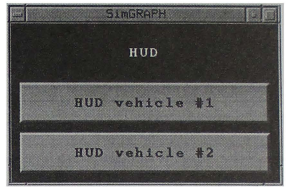


Figure 22 - HUD Selection Window

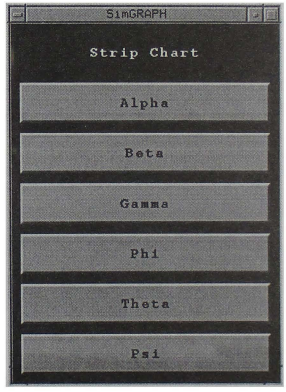


Figure 23 - Strip Chart Selection Window

The Close Graphic Objects Button on the Main Option Window (Figure 19) allows the user to access the Close Graphic Objects Window (Figure 24). The window presents the user with a list of the Graphic Objects that

are currently open. The user can select one of the Graphic Objects and then select the Close Object Button. The Graphic Object will be removed from the screen.

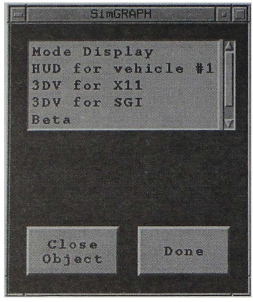


Figure 24 - Close Graphic Objects Window

Selection of the User Configuration option from the Main Option Window (Figure 19) causes the User Configuration Window (Figure 25) to be presented to the user. This window allows the user to save and load configuration files. These configuration files consist of the Graphic Objects that are open and their positions on the screen.

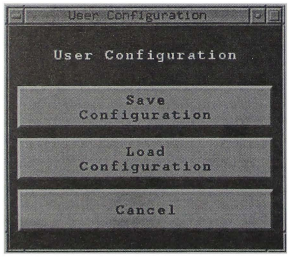


Figure 25 - User Configuration Window

The Replay Button on the Communication Selection Window (Figure 18) allows the user to examine data from a previous SimGraph session. The data may be viewed either forward or backward at various speeds. All the information for the displays will be loaded from a data file instead of being transmitted from the simulation program. After the user selects a data file, the program replaces the normal menu system with the Replay Control Window (Figure 26).

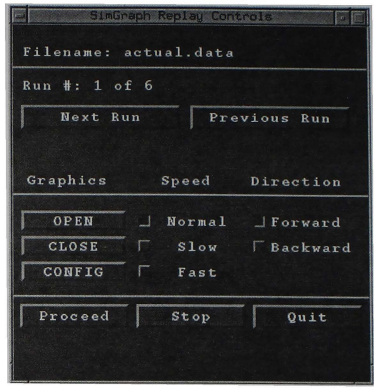


Figure 26 - Replay Control Window

If the user wishes to open Graphic Objects to be viewed during the replay, the Open Button on the Replay Control Window must be selected. Activating the Open push-button causes the Open Graphic Object Box (Figure 27) to appear. The user may then select which Graphic Objects will be displayed on the screen.



Figure 27 - Open Graphic Object Box

To close an open Graphic Object, the Close Button on the Replay Control Window must be selected. Activating the Close Button causes the Close Graphic Object Box (Figure 28) to appear. The Graphic Objects

that are currently open will be shown in the scrollable list area.

Selecting the Config Button from the Replay Control Window (Figure 26) will cause the User Configuration Window (Figure 25) to be displayed. The remaining buttons on the Replay Control Window dictate how the data is to be replayed. If Normal is selected, the data will be replayed in real-time, if possible. If Slow is selected, the data be shown at one fifth of real-time speed (i.e., the time between frames will be multiplied by five). If Fast is selected, the data will be shown as fast as possible. The next panel of buttons is used to indicate replay direction. Forward and Backward allow the user to move through the data in the selected directions. The bottom panel of buttons allow the user to begin playing data (Proceed), halt the playing of data (Stop), and exit SimGraph (Quit). The currently selected option will be highlighted in red.

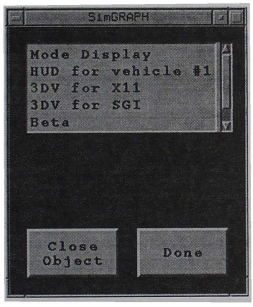


Figure 28 - Close Graphic Object Box

Conclusions

The increasing complexity of modern flight simulation has created the need for tools to aid in the visualization of the vast amount of data that is produced. SimGraph was created to aid users in data interpretation and allow for rapid modification to handle future requirements. The modular design of the program allows the user to pick and choose from a library of graphical displays that will meet the needs of the research being conducted.

The Graphic Objects discussed were used to evaluate a prototype SimGraph. The first production model of SimGraph is to be utilized by the Transport Research Facilities at NASA Langley Research Center. Several new Graphic Objects are currently under production for this simulation.

While this system was specifically designed to work with the flight simulation software at NASA Langley Research Center, it is not exclusive to that research environment. It is possible to use the framework that was developed and create new Graphic Objects to adapt the software to individual environments. The authors welcome inquiries for additional uses of the system.

References

1. Crawford, D. J.; Cleveland II, Jeff I. "Langley Advanced Real-Time Simulation (ARTS) System", Journal of Aircraft, Volume 25, Number 2, February 1988, Pages 170-177.

2. Edel, Mark. "Histo-Scope Plotting Widget Release 4" ftp://ftp.fnal.gov/pub/histoscope/v4_0/histosgi.tar (4 Feb. 1995)

3. OpenGL Architecture Review Board, OpenGL Programming Guide The Official Guide to Learning OpenGL, Release 1. Reading, Massachusetts, 1993.

4. Dare, A.; Burley, J. "Pilot/Vehicle Display Development From Simulation To Flight", AIAA 92-4174, August 1992.

5. O'Reilly, Tim; Nye, Adrian X Toolkit Intrinsics Programming Manual, Motif Edition, Volume 4, November 1992.

6. Rainey, Riley. "ACM Version 4.8. The Aerial Combat Simulation for X11" ftp://ftp.netcom.com/pub/ra/rainey/acm/acm-4.8.tar.gz (10 Apr. 1997)

Model Reuse for Software Development and Testing: The Application of Common Interfaces to Support Variable Fidelity Models

William J. Bezdek *
McDonnell Douglas Corporation
St. Louis Missouri

Thomas A. Halley
McDonnell Douglas Corporation
St. Louis Missouri

Parl C. Hummel
McDonnell Douglas Corporation
St. Louis Missouri

Abstract

It is desirable to use object-oriented design techniques and common interfaces which map closely to the "real world" to achieve maximum reuse of aerospace models for both development and testing. The model development process should support and provide models at different levels of detail (fidelity), so the designer/user/analyst/tester can select the version needed for the activity being performed. The simulation should also be flexible enough to allow the models to change from one fidelity level to another during execution to minimize overall runtime. There are several graphical tools and techniques available that support object-oriented design by the subject matter experts including Rational Rose, JMASS, ISI MATRIXx, and i-Logix Rhapsody. Models should be developed and tested to verify they can operate in a simulation using common interfaces and that the interfaces can be modified/expanded to allow the different "levels of detail" to be used. Bandwidth trade-off vs. fidelity and the capability to switch models during execution will be discussed. The conclusions will show the advantages of using object-oriented design and "real world" interfaces over traditional methods and the affordability and maintainability improvements possible. Future applications include applying the same processes and tools to reusable aircraft avionics software.

Introduction

Software model building has progressed from specialized models designed and coded for each application to a concept where reusable models are utilized within multiple applications. The use of object-oriented design techniques and common interfaces that closely approximate "real world" objects provides the capability to achieve maximum reuse of aerospace models for both development and testing.

The model development process should provide models at different levels of detail (fidelity), so the designer/user/analyst/tester can select the version needed for the activity being performed. The models should be adaptable to different types of simulations including:

- Constructive,
- Virtual, and
- Live.

The model development process should be flexible enough to allow the user to select the fidelity needed for that series of runs, be able to revalidate the model, and be able to track the validity of the model back to the source. The model development tools should provide a user-friendly, graphical way to develop and store the models to achieve maximum reuse at minimum cost.

Previous Experience

In the past, many simulation models were written in FORTRAN and 'C' and adapted to the specific requirements of that simulation. These models were modular in nature and constructed using subroutines/functions to approximate "real world" boundaries: (e.g., the control system, the weapons, the avionics, the engine, etc.). These were connected using global interfaces without information hiding. These models often had built-in simulation features, such as:

- the ability to initialize, stop, or hold model state,
- the maximum number of models available were built into array sizes (i.e., the number of weapon stations on an aircraft), and

*Senior Member

Copyright © 1997 by the American Institute of Aeronautics and Astronautics, Inc. All rights reserved.

- the data was often hard coded into the program.

Other simulations had features built into the models that required a lot of set-up, instead of keeping the data separate from the model.

The government has several constructive level models, like TAC Thunder[1], EADSIM[2], TAC Brawler[3], Suppressor[4], and ESAMS[5] that have many of the same problems. These models have been described as stovepipe models[6], in that they don't interact with each other or work together. These simulations are being replaced with new object based modeling and simulation systems, like Joint Warfare System (JWARS)[7], Joint Simulation System (JSIMS)[8], and Joint Modeling and Simulation System (JMASS) [9].

The government and industry have been using DIS[10] for several years to support networking between separate simulations and is in the process of developing and fielding a networking concept called the High Level Architecture (HLA)[11] to expand the linking of simulations. HLA will be used to allow all simulations to be linked over the next few years.

Model Design

To maximize reuse of Modeling & Simulation (M&S) models, objects, and object components, the model designer must understand what development and testing activities these components need to support and where these activities will take place. That determines whether one model will support all anticipated activity or whether a family of models with different fidelity levels and different resource constraints should be developed. Whenever possible, new or redesigned components should be used to leverage designs presently available in one of the legacy programs.[12]

A reuse library should support storing the design as well as the source code, and the graphical design tools should contain a reverse engineering tool to allow legacy models to be updated/converted to the standard format. The reverse engineering tool should also allow changes to be made to the models at the source code level and automatically have them added back into the graphical model. Rational Rose has that capability; i.e., if another attribute is needed, it can be added to the source code and becomes part of the graphical model. The automatic code generator should also create protected areas that allows the software designer to add code segments that become part of the model and are saved when the model is regenerated. JMASS, for example, has a capability called "spin and weave" that allows the software designer to add behavior code in special areas that are retained when the Model Component Development Tool (MCDT) regenerates the model.

The Modeling Guidelines shown support the development and reuse of models to improve affordability. The guidelines recommend that:

- Modeling objects conform to a common Software Structural Model (SSM) to maximize object compatibility and interchangeability without inhibiting model efficiency or model developer creativity,
- Model development be done by domain experts as opposed to strictly software experts,
- Tools be used to support development of model components which comply with the architectural standards (i.e., developers should not have to worry about the model's target language, only about designing it with visual engineering tools),
- Models be designed for maintainability over an anticipated life cycle of 25 to 30 years,
- The Verification, Validation, and Accreditation (VV&A) process be done in accordance with DoD and Commercial requirements and test cases/results included with the model,
- Environmental modeling be done to support analysis throughout the frequency spectrum (RF, IR, EO, visual),
- Multiple languages be supported (FORTRAN, C, C++, Ada83 and Ada95),
- If existing "legacy" model is used, provide a wrapper to conform to "real world" interfaces and use a procedure call to hide the implementation details,
- Models be developed to support a distributed processing environment,
- A reuse library be established and supported by management to encourage model reuse, and
- Models stored, as much as possible, with all design details, including requirements, algorithms, documentation, to allow different tools to be used.

Level of Fidelity

Three "levels of fidelity" are recommended for most simulation applications including:

(1) Analytic/Constructive models which provide state information and basic cross section for different angular relationships,

(2) Dynamic models that have the capability to interact with the on-board avionics and other systems including the pilot. Many manned simulations fall into this category and are required to maintain real time while using "real world " interfaces, and

(3) Emulative models that are capable of operating in a detailed electronic warfare environment. This

category tries to, as nearly as possible, provide an exact software replica of the hardware or real system.

The relationship between man and machine is often used relative to model fidelity as shown on Table 1.

Table 1 Man/Machine Interface

Fidelity	Content			
	Man		Machine	
	Sim.	Real	Sim.	Real
Analytic	✓		✓	
Dynamic		✓	✓	
Emulative	✓		✓	✓

For example, the different levels of fidelity for an aircraft are shown on Figure 1.

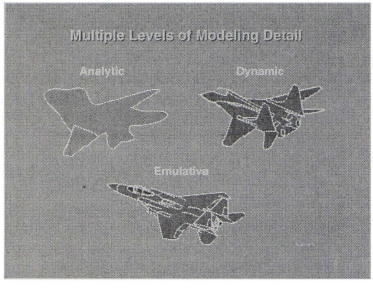


Figure 1 Model Level of Fidelity

Analytical Models

Analytic models are often used in constructive simulations, which require basic information on the model, but in a way that minimizes computer resources, so many models can be instantiated. For example, using MATRIXx to analyze the effects of aerodynamics and loads on the airframe, a very detailed 6 DOF model may require as many as 20,000 source lines of code (SLOC). If this detailed model is designed and executed in MATRIXx[13], within two or three days the user can:

- compute the trim point,
- generate a linear model about the trim point,
- compute the frequency response of the system, and
- perform a gradient search for the coefficients of the equivalent system's transfer function

all within MATRIXx.

Mission software testing can often be adequately accomplished with an equivalent system model represented with as few as 200 SLOC. The 100 to 1 reduction in model size reduces compile and execute times from minutes to seconds. Consequently, a mission software development and testing process that uses the MATRIXx equivalent system model avoids the time and cost associated with independently developing and maintaining an air vehicle model.

Dynamic Models

Development simulators are normally used to evaluate and test new concepts, including:

- aerodynamics,
- flight controls,
- engines,
- avionics,
- sensors,
- threat tactics, and
- weapons

in real time simulations with project personnel and pilots, and are used to justify modifications for the next iteration or funding cycle of the weapon system. This type of simulation is also used extensively for pilot demonstrations. With the advent of faster processors, the model used for manned simulation is often very close to a handling qualities model.

The models being developed need to be able to support and interface with the dynamic executive structure to allow the aircraft, weapons, sensors, visuals, etc. to communicate with each other.

Emulative Models

Emulative models must often operate at very short update intervals to support operations like pulse-to-pulse radar and electronic countermeasures.

The emulative model is not normally required to operate in real time, but must support the same message passing / interface used in the other models with additional data to support the higher fidelity calculations. Bandwidth may often increase from approximately 160 variables, 20 times per second to 1250 variables, 200 times per second. This 100 fold increase in data would overwhelm an analytic or dynamic model; therefore the emulative model should be run off-line and generate tables which can be used in analytic/dynamic models, but still have traceability back to the detail model.

Common Interfaces

The way to make models interchangeable is to pay special attention to the interfaces that the models use.

In particular, it is important to use inheritance of a common interface for the basic vector/matrix/body classes as shown in part on Figure 2.

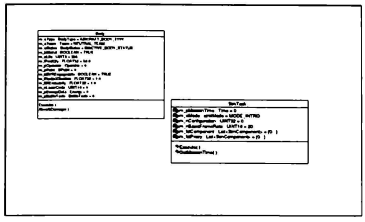


Figure 2 Common Interfaces

The common interfaces can be determined from previous experience with distributed simulations, providing base classes that produce common results, like matrix and vector information including state information for moving bodies, location for targets and fixed objects, etc. The exact representation of the classes needs to be investigated in more detail. As a baseline, the information needed for all models including the constructive model will be listed first. The dynamic/emulative models will append additional messages, if needed.

In many cases the data needed for an emulative simulation is compiled and published by the subject matter experts and known as the models are developed. These interfaces may not be extensions of the analytic and dynamic models, but it would be desirable in the future.

Graphical Design Tools

Recently, the use of Ada95 and C++, along with object-oriented design tools and techniques, has had a large effect on reusability and extensibility of simulation models. There are several commercial graphical tools and techniques, like Rational Rose[14], ISI MATRIXx[15], and i-Logix Rhapsody[16] available to help support object-oriented design by the subject matter experts. In addition, there are several government architectures to help provide reusable, standardized models, like JMASS and others. A typical set of tools used for object-oriented, graphical software development is shown in Figure 3[17]. This typical set includes a/an:

- requirements tool to capture model requirements,
- graphical display tool using open systems,
- graphical object/connection drawing tool,
- automatic code generator,
- configuration management system,
- test case tracking system, and
- debugger.

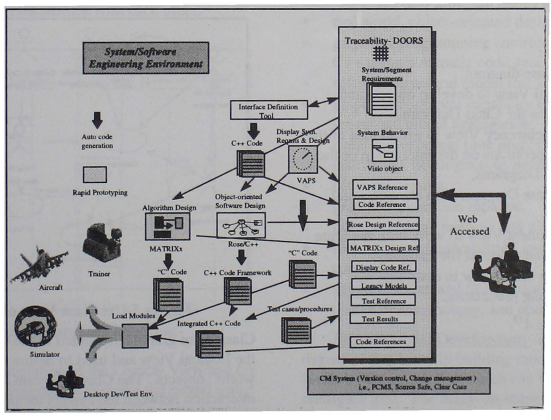


Figure 3 Software Processes and Tools

With graphical tools there is often a cost of ownership, difficulty in model transportability, and the capability to modify a model is significantly impacted if the person wanting to reuse the model does not have the same tool. To provide a more robust model library, it is desirable to store the model design as well as the complete source code for the model, including a description of the model behavior that can be readily used by any tool that supports automatic code generation. Storing the model design also allows models to be independent of programming language and allows different tools to be used with the models.

For example, JMASS provides an intermediate ASCII file called the "description file" with each model that describes the inputs, outputs, attributes, and operations. Rose has an ASCII "CAT" file with similar, but different, information that goes along with the model file. In addition, the tests used to validate the model should be stored with it to allow it to be retested in the future.

Rational Rose

Rose, one of the most complete commercial, object-oriented, graphical software tools available now, was developed by Rational Corporation and is operational on several platforms including PCs. Rose uses the Unified Modeling Language (UML)[18] developed by Jim Rumbaugh, Grady Booch, and Ivar Jacobson, and supports a concept called the "four plus one view" of the problem.

The developer can:

- build a Use Case diagram,
- use the Logical View to develop objects and relationships on the Class Diagram,
- apply the Concurrency View to get an idea of the sequence of operations on the Interaction or the Collaboration Diagrams,
- expand the Class Diagram with attributes and operations,
- use the Component View to work with the State Diagram and the physical file mapping,
- use the Deployment View to distribute the processes among processors,

as shown on Figure 4[19].

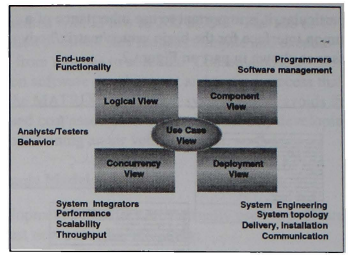


Figure 4 Four plus One View

The application of the Use Case View is being used for requirement analysis and captures the functional requirements of the system. Rose provides a way to then map requirements into an object model. Use Cases can be used to bridge the gap between the analyst, customer, application developer, and help define the tests required.

For the mission used as an example throughout this paper, a commander (actor) selects the target, aircraft type, and pilot for the mission. The pilot selects the weapons desired and flies the mission. On the "enemy" side, the commander, pilot, and SAM controller try to keep the target from being destroyed as shown on Figure 5.

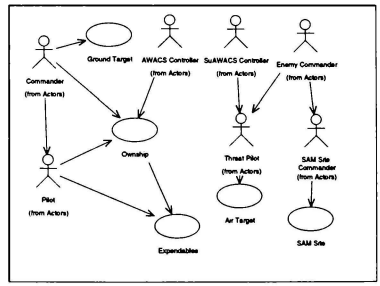


Figure 5 Use-Case View of Mission

Classes are graphically added to the Class Diagram in the Logical View and used to define abstract "real world" objects. The Class Diagram also shows the interaction between the classes including inheritance and associations as shown on Figure 6.

In this case, the Missiles, Bombs, and Guns are "a kind of" weapon and inherit from the Weapons class. The

Avionics, Airframe, and Weapons are "part of", or aggregates of the Ownship. There is also an association between the Target, Ownship, and SAM_Site, which allow messages to be sent between them. The figure also shows which packages the classes belong to that provides a way to group similar classes and objects.

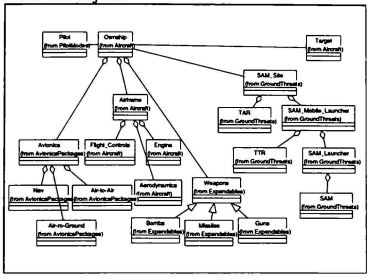


Figure 6 Class Diagram

The Interaction and Collaboration Diagrams are subsets of the Concurrency View. The Interaction Diagrams show how the objects, defined in the Use Case analysis, and the classes, defined on the Class Diagram, interact as part of a scenario, as shown for a small segment on Figure 7. The Sequence (Interaction) Diagram is also used to determine additional objects / classes that are needed and their relationships with each other.

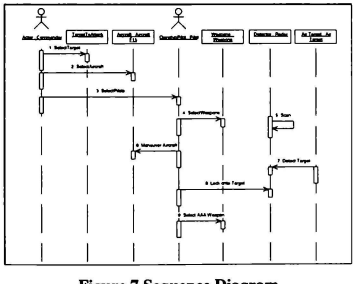


Figure 7 Sequence Diagram

Classes are the foundation of the Logical View and are used on the Rose diagram to describe the model using object-oriented design techniques. The Class Diagram can now be expanded to add the attributes and the behaviors for each class. Rose provides an automatic code generator, so when the classes are described, they can be automatically generated. The inheritance shown on the diagram causes the code generator to add constructors and destructors and the associations can be linked to generate a complete object-oriented model. A partially expanded class diagram is shown on Figure 8. The user is still responsible for adding the algorithmic code into the automatically generated source code.

The Component View is often used to view the state diagram of the system. Since Rose allows the user to switch between the different views of the problem, the user may adapt the sequence shown here or his/her own preference. The Component View also is used to allocate classes and objects to modules in physical systems, show dependencies, and compilation order.

Lastly, the Deployment View is often used to assign tasks to processors, map a distributed network, or other communications features. It shows the configuration of runtime processing elements.

JMASS

The Joint Modeling and Simulation System (JMASS)[20] is a government sponsored, object-based M&S system. JMASS supports the development and execution of model entities from complete aircraft, radar, and weapon systems (which JMASS refers to as players) to multi-team distributed simulations that can support a many-on-many engagement. The methodologies used by JMASS to support a wide range of M&S needs include:

- real world, object-oriented design,
- graphical programming environment for Ada or C++ using automatic code generators,
- software structural model that supports reuse,
- graphical configure capability to support scenario set-up and model aggregation,
- event-based or real time processing on a distributed, heterogeneous network,
- batch processing,
- graphical execution monitor,
- post-processing analysis and playback,
- library of verified/validated model entities and components,
- support on Sun and SGI platforms, and
- user manuals, training, and operational support.

JMASS Model Development -The user has a high degree of flexibility in building models with different levels of detail, some of which can be selected at configure time. For many-on-many engagements, as in other simulations, it is important for all models to have

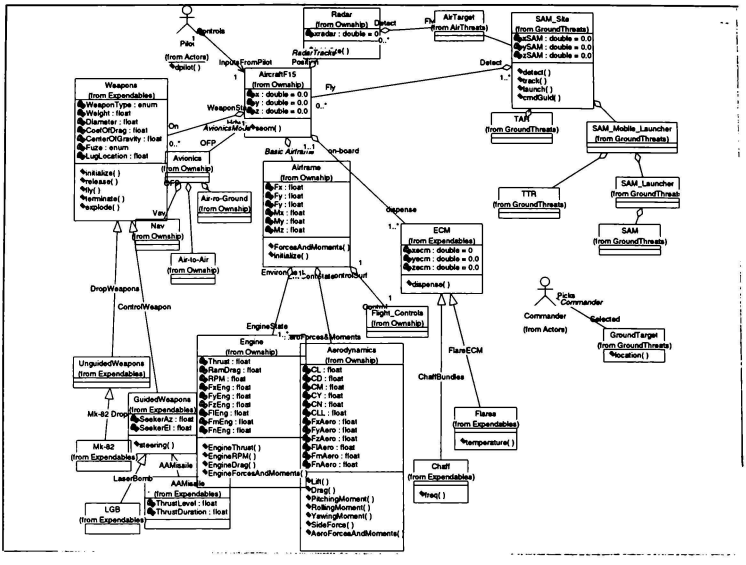


Figure 8 Rose Class Diagram of Mission

common interfaces so that another models with a different level of detail can easily replace the existing model. JMASS uses ports to allow a model to interface with other models and supports an Application Program Interface (API) to allow modelers to design models which can be integrated into a simulation.[21]

Dynamic Level of Detail Control - Users may want to dynamically switch model entity levels of detail during a run. This is possible with JMASS by compiling and building teams of players that all contain the levels of detail needed using one of several methodologies. A previous paper discussed downgrading the individual players to assemblies and adding a new player broker which interacts with the user/simulation which can dynamically select the level of detail desired. This was done using a port to the player broker that receives inputs on the version of the player to use and calls the appropriate assembly. A multi-level target, capable of running at varying levels of fidelity, was developed using JMASS in 1996.[22]

JMASS Target Model - An example of an object-based model from JMASS is the Generic Aircraft Target Model (GATM), developed to provide an air target for the development of on-board systems like an Electronic Countermeasures system, etc. The top-level graphical diagram for GATM is shown on Figure 9.

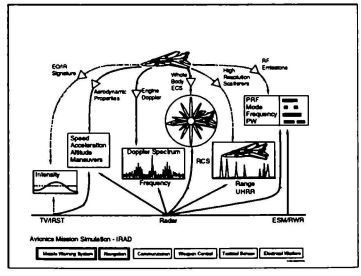


Figure 9 GATM Target Model

MATRIXx

MATRIXx is a mathematical modeling and analysis tool commonly used in the design and evaluation of flight control systems. The MATRIXx tool palette is used to graphically design models of both the flight control laws and the air vehicle model with six degrees of freedom (DOF). Control surface actuators, aerodynamic and propulsion forces and moments, mass properties, the atmosphere, and the equations-of-motion (EOM) models that govern rigid body motion using standard input and output lists can be interfaced using GUI dialog windows. A typical MATRIXx graphical output is shown on Figure 10. It contains both logic and user code blocks that represent code to be linked with the MATRIXx code.

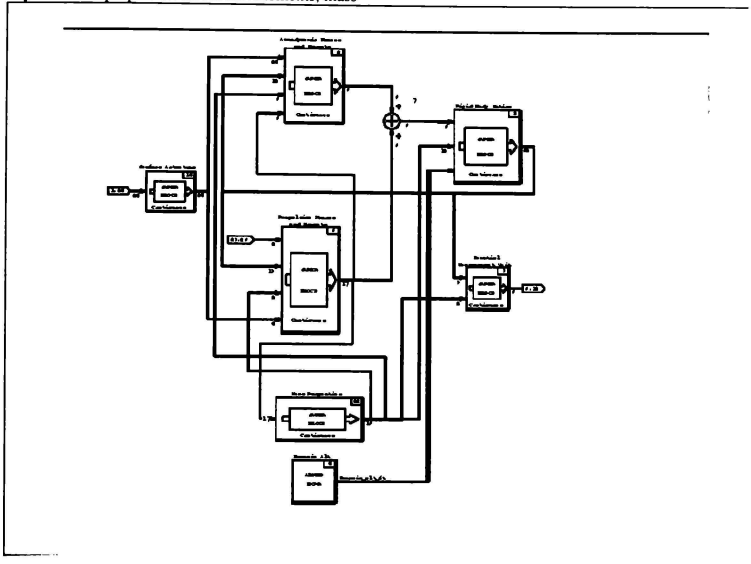


Figure 10 MATRIXx Graphical Model

While F-15, F/A-18, and AV-8B air vehicle models all conform to the architecture of the generic air vehicle model and have MATRIXx representations, the most detailed (a.k.a. handling qualities) models make extensive use of a user code block (UCB) in the aerodynamics, propulsion, and mass models. A UCB is represented graphically in the MATRIXx model and employs a graphical dialog window to connect its inputs and outputs to the rest of the model, but must be bound to the user's Ada, FORTRAN, or C code prior to executing a simulation. This is an attractive capability that can expedite the transition of legacy models to MATRIXx, supports debugging the code, and verifying the output off-line.

In addition to the graphical system design capability, MATRIXx also has a command line user interface to select additional analytical tools. The availability of a large number of analytical tools within MATRIXx encourages the standardization of models and methods. The result is increased reuse of existing models and less effort expended creating them.

A key feature is MATRIXx's ability to automatically generate code and document it. Typically, design and analysis groups identify software requirements that must be satisfied during the software development and testing activity. Unfortunately, requirements identified by the analyst are subject to misinterpretation by the software developer. Giving the analyst the means to

create a design satisfying the requirements and automatically generating code from his design circumvents communication breakdowns and intensifies the developer's knowledge. Although the automatically generated code tends to be less efficient than code generated manually, experience has shown this approach yields productivity improvements of 500% while increasing the size of the code by less than 30%.

Rhapsody

The i-Logix Rhapsody[23] Object-Oriented Analysis, Design and Implementation Tool is a new product similar to Rational Rose and provides development tools to meet the needs of real time embedded applications. Rhapsody produces complete production quality code from high level designs and graphically animates execution behavior based on the State Diagram representation of Dr. David Harel. It allows software designers to spend less time coding and debugging and more on design and analysis.

Other Methodologies

HLA - The high level architecture (HLA) is a government sponsored system that provides a communications infrastructure to facilitate interoperability between different simulations and the ability to "federate" existing simulations in order to improve reuse. HLA uses modular simulation components constructed with well-defined interfaces. HLA provides interoperability within simulations, among simulations of a federation, or across functional communities. HLA supports a broad range of functional areas including training, contingency planning, analysis, and acquisition. Applicable simulations involve software only representations, man-in-the-loop simulations, and live components. [24]

JWARS - The Joint Warfare System (JWARS) is a government sponsored system that will provide the infrastructure to simulate large campaign simulations used for acquisition and planning. Operational commanders and analysts can use JWARS to model the performance and predict the effectiveness of a wide range of weapon systems and threats. [25]

JSIMS - The Joint Simulation System (JSIMS) is a government sponsored system that will be a single, seamlessly integrated simulation environment. It will include a core infrastructure and mission space objects maintained in a common repository. These can be composed to create a simulation capability to support Joint or Service training, rehearsal, or education objectives[26].

Testing

It is important to use a spiral design methodology, so the modeling and simulation components that support system development and testing activities can be shared by several development and testing environments, as shown on Figure 11.

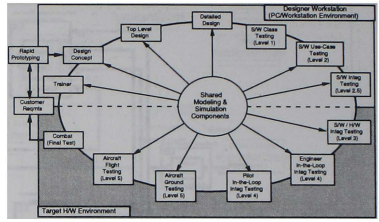


Figure 11 Testing Cycle

Design and initial testing of embedded software will take place on Designer Workstations in the PC/Workstation Environment.

Once customer requirements are well understood and prototype design concepts acknowledged, responsibility for developing each design concept is assigned to an Integrated Product Development (IPD) team, which is composed of representatives from several development and testing disciplines. The IPD team proceeds from prototype design concept to detailed design in the Software Development Facility (SDF), which resides on the standard PC/Workstation configuration.

Initial software testing (class testing) also is accomplished in the SDF by augmenting the tools necessary to facilitate software design and construction with tools that facilitate object-oriented class testing. Intermediate software testing (i.e., cluster testing and software integration testing) is accomplished in the Desktop Test Environment (DTE), which also leverages the standard PC/Workstation configuration with additional tools tailored to support such testing activities.

At this point, a transition is made to test on the target hardware. Traditionally, software / hardware integration testing of the Mission Computer subsystem is accomplished in a Software Test Facility (STF). To verify that functionality has not been lost in the transition from host to target hardware, some of the

testing accomplished on the PC/Workstation configuration can be repeated using the same tools used to support class and cluster testing in the DTE. Additional tools tailored to support testing the whole subsystem are used for the most part.

The next step is to test how well subsystems work together. Traditionally, this has been accomplished in two separate facilities. If the evaluation was done from an engineering perspective, then such testing was performed in the System Integration Facility (SIF). But if the evaluation was done from a pilot's perspective, then that testing was performed in a Manned Flight Hardware Simulator (MFHS). The distinction between these two testing environments is now blurring. In the future, it will be quite possible to run some pilot evaluations in the SIF, which will have some Out-the-Window (OTW) display capability, which is necessary to support that testing. Only when the pilot wishes to evaluate how the avionics system provides feedback on aircraft flying qualities is it essential to perform tests in the MFHS. Tools to support testing in these environments are similar to those used to collect flight test data.

On-aircraft ground testing and flight testing is conducted to ensure that customer requirements have been satisfied in the production aircraft / weapon system and that the aircraft is capable of performing all assigned missions when flown by operational test pilots. Flight instrumentation systems monitor the interaction between subsystems as the pilot performs tests that are representative of various combat phases, and other airborne and ground-based monitoring systems establish actual performance of the aircraft / weapon system under test.

Trainers are also designed to closely replicate how the aircraft / weapon system will perform assigned missions, but now testing tools are designed to measure the performance of the trainee rather than the aircraft / weapon system. The trainer itself very much resembles a high fidelity Development Simulator and shares many of the same modeling and simulation components used in that environment.

Conclusions

The use of graphical tools to generate new and updated models has proven to be very effective, if the developer designs the models around reuse and common interfaces are used. It is important to store the design, as well as the code, for the models in a reuse library. The graphical design tools should contain a reverse engineering tool to allow legacy models to be updated / converted to the standard format and allow changes to be made to the models at the model design level and automatically added back into the code.

Existing models should use wrappers to allow the models to be used. This allows existing model to take advantage of "real world" interfaces, and allows a "real world" hardware and software models to be substituted. The start-up costs vs. the long-term maintainability cost of using legacy models should be addressed early. Even though setting-up a wrapper for an existing model is often faster and easier, maintaining the model could be much higher.

To maximize M&S component reuse, components should be designed in a modular fashion, with well-defined interfaces. Design logic/rationale of shared components should be maintained in a reuse library to facilitate valid reuse decisions and contain:

- object based designs that support component reuse,
- a repository of models and their components, and
- documentation/testing.

Future Applications

Legacy Models

The models for the simulation can be converted in two ways:

1. Using wrappers on existing legacy models with common interfaces and
2. Using new models at different "levels of detail" based on reusing the behavior code of the legacy models with object-oriented tools and processes.

A comparison of the execution and development time using the two methodologies will provide lessons learned, as well as working models, for a simulation. Generating interfaces for different "level of detail" models will provide a baseline of how to change from one model detail to another during execution.

Avionics Design and Testing

The architecture used for object-oriented models can also be applied to flight hardware in building embedded software for flight. Many of the same design principles apply as well as the desire to be able to develop on one platform and then move the code to different target hardware.

In addition, it would be useful for the software developed for a simulation to be the same as that used in flight. There are several additional features that are needed for simulation software, like initialization in flight, slewing to different areas of the world, stopping,

and starting the simulation, that are not normally included in flight software. With the object-oriented principles and the use of polymorphism, a flight hardware load could be very similar for the development, simulation, trainer, and flight hardware version. If successful, it would have a significant impact on overall program affordability.

---

[1] TAC Thunder Overview Briefing, 1995.
[2] US Army Space and Strategic Defense Command, EADSIM, 1995.
[3] TAC Brawler User Manual, 1994.
[4] Suppressor User Manual, 1993.
[5] ESAMS Technical Capabilities Sheet, 1995.
[6] Gen. Case Briefing, AFXOM, 1994.
[7] JWARS Briefing, May 1996.
[8] TRD 041896, Joint Simulation System (JSIMS) Joint Program Office (JPO), "Technical Requirements Document", 4/19/96 (Draft)
[9] J-MASS ORD, March 1996 (Draft)
[10] Beavin, W., Emrich, T. and Bezdek, W., "DIS Interface Using JMASS", NAECON, May 1995.
[11] HLA Briefing, Dr. Judith Dahlman, Dec. 1994.
[12] W.J. Bezdek, "Preliminary Modeling Guidelines", McDonnell Douglas Corporation, 28 February, 1997.
[13] Integrated Systems, Inc., System Build User's Guide, 1996.
[14] UML for Rational Rose 4.0, Rational Software Corporation, 1996.
[15] Integrated Systems, Inc., System Build User's Guide, 1996.
[16] Toni Malins, Rhapsody Object-Oriented Analysis, Design and Implementation Tool, I-Logix, Inc., 1997.
[17] Lori Kilpatrick, "System/Software Engineering Environment", McDonnell Douglas Corporation, March 1997.
[18] Unified Modeling Language (UML), Rational Corporation, 1996.
[19] Mark Schulte, Rational Rose Training Class, McDonnell Douglas Corporation, March 1997.
[20] W.J. Bezdek & C. S. Herr, "Use of JMASS for a Many-on-Many Engagement Model", AIAA-96-3492, July, 1996.
[21] JMASS User Manuals - Volume IV - Model Developer Manual, Appendix A, JMASS-GS-3.0a, , 15 April 1996.
[22] W.J. Bezdek & E.A. Reidelberger, "Vehicle Interactive Digital Pilot using JMASS", AIAA 96-3494, July, 1996.
[23] i-Logix Rhapsody Briefing at McDonnell Douglas, May, 1997.
[24] Dr. Judith Dahlman , HLA Briefing, , Dec. 1994.
[25] JWARS Briefing, Washington D. C. May 1996.
[26] TRD 041896, Joint Simulation System (JSIMS) Joint Program Office (JPO), "Technical Requirements Document", 4/19/96 (Draft)

AIAA-97-3800

APPLICATIONS OF DISPLAY PROTOTYPING AND REHOSTING TOOLS TO THE DEVELOPMENT OF SIMULATOR, FLIGHT TRAINING DEVICE, AND COCKPIT DISPLAYS

Paul A. Bennett
Manager, Technical Marketing
Virtual Prototypes Inc.
Montreal, Quebec, Canada

Abstract

Recent trends toward the use of 'glass cockpit' avionics suites in cockpits and in the simulators and flight training devices that serve as 'live' cockpit models have increased the burden of software development on avionics manufacturers. Increasingly, such software-oriented problems are being solved through the clever application of advanced visually oriented software development tools. Such software tools are applied to the design, prototyping, testing, and deployment of avionics displays. The present paper discusses successes achieved with one such tool suite. Developed by Virtual Prototypes Inc. of Montreal, Quebec, Canada, VAPS and its companion C Code Generator (CCG) are specifically targetted at the development of avionics displays. VAPS allows users to develop avionics displays in a point-and-click fashion without resorting to programming. Having developed an avionics display, it can be iteratively animated, tested with real users, and modified until both designers and eventual users are satisfied that an optimal design has been achieved. Avionics displays can be rehosted to such computing environments as workstations, PCs, and ruggedized embedded systems in a matter of minutes through a process that turns the specification for the display into executable code. This process is described as are its outcomes.

Introduction

Increasingly, avionics manufacturers are replacing electro-mechanical avionics instruments with 'glass cockpit'-based instrumentation. Such avionics suites feature representations of aircraft flight parameters on a display device such as a Cathode Ray Tube (CRT)-based monitor or on some form of flat panel display, hence the term 'glass cockpit'.

While CRTs and flat panels provide the physical output medium for glass-cockpit avionics displays, they are actually driven through some form of on-board computer. Using computers pre-supposes programming them. Until recently, that's just what avionics manufacturers have had to do: they have had to program avionics displays for glass cockpit environments by hand.

The Challenges

Programming avionics displays for glass cockpit environments is a challenging occupation. It tends to take a long time to accomplish and to be error-prone. As well, a variety of commercially oriented considerations impinge on the process, sometimes impacting technical decisions in strange ways.

The resources challenge

The traditional problem with building avionics displays has been that it takes a lot of people a long time to accomplish the process.

Copyright © 1997 by Virtual Prototypes Inc.
Published by the American Institute of Aeronautics and Astronautics, Inc. with permission.

Designing

Displays would first be specified on paper using diagrams and plenty of textual description. This would create a situation where some aspects of the design might be specified incompletely or ambiguously.

Hand-coding

Once the design was complete, a software implementation of the display would be designed and a team of programmers would implement the design either in a high-level language such as C or Ada, or in a low-level language such as the display processor's assembly language.

In general, programming tends to be a frustrating and error-prone process. These attributes of the programming process usually translate directly, though not linearly, into the time required to implement a system.

As well, the difficulty of programming software such as avionics displays is exacerbated by the difficulty of using a particular programming language. Thus, programming displays in an assembly language will generally be more difficult and error-prone than would programming in a higher- level language like C or Ada. This principle could be extended to state that if one had a higher level avionics programming formalism than either C or Ada, the difficulty level and error-proneness, as well as the time taken to implement avionics displays, should decrease.

The time taken by an avionics development team to hand code their avionics products gives the market for such devices plenty of time to evolve. Once the development team has produced a product and passed it along for testing, the testers' expectations of what constituted useful or 'state-of-the-art' products might have had a year or more to develop. Regardless of the market's evolution, design flaws might only be uncovered when the process of building the product was nearly finished.

Sometimes, the software team would find that the design specified by the avionics designers at the beginning of the process could not be adequately implemented using the hardware available to them. The whole design process would have to be revisited until a 'more reasonable' design was settled upon.

Taken together, these factors tended to mean that programming avionics displays would require a lot of people a long time using an error-prone process to develop avionics displays which were probably out of date by the time they made it into an actual glass cockpit.

Avionics manufacturers would design and implement new computing technologies, often at huge expense, in order to implement acceptable avionics displays. Each manufacturer's implementations would be different, and shifting a large development contract from one avionics manufacturer to another would often involve prohibitive costs when the alternate manufacturer would have to re-invent proprietary technology.

These prohibitive expenses tended to reduce competition among producers of avionics, prompting governments, among others, to seek ways of reducing these expenses.

The COTS challenge

On the one hand, governments have insisted on cost reductions when making acquisitions of military avionics products. On the other hand, governments have insisted that avionics manufacturers adopt strategies involving Commercial Off-The-Shelf (COTS) strategies wherever possible.

While insisting that COTS strategies be used in the design of avionics, the demands being placed on developers of avionics displays have in no way diminished. Avionics displays have become more challenging to build, as the study of human factors and of ergonomics has progressed.

Similar forces have been at work in the commercial arena where developers of avionics for non-military uses increasingly specify very demanding COTS solutions to be delivered inexpensively over short time spans.

Avoiding design lock-ins

The last number of years have seen substantial increases in the processing power available in a variety of computing packages. In parallel, increasing demands are placed on the computational infrastructure. Indeed, computing platforms have developed so rapidly in recent years that it has now been realized that it may not be possible to predict, at the beginning of an avionics project, the nature of the computing platform on which it will eventually be flown on an aircraft. Software portability has emerged as an over-riding software requirement for avionics products after many avionics projects have had to be redeveloped from the ground up in order to accommodate new hardware. Customers refuse to be locked into a specific computing environment.

The Technical Challenge

The need to produce better avionics displays faster at a lower cost for a possibly unknown computing platform using COTS equipment and products whenever possible translates into some fairly demanding technical requirements for developers of modern avionic displays.

Performance

One of the most important technical criteria for avionics displays is performance. The term 'performance' means that information should be displayed constantly, smoothly, and in a timely fashion, according to the nature of the display, the data involved, and the type of aircraft involved. High performance displays are used to create a one-to-one relationship between the representation of flight or other aircraft data on the screen and the physical reality of the aircraft in the aircrew's minds. Specific performance requirements seem to vary between approximately 12.5 HZ and 30 HZ.

As well as creating high-performance displays, there is a requirement for a performance buffer. The computing environment must be able to do even more display computations should the need ever arise, and to act as a safety net ensuring that the display system can never run out of resources even in the most demanding circumstances.

Flexibility

As discussed earlier, a wide range of avionic displays must be produced and deployed easily. As well, it must be possible for the displays to be run on a wide variety of computing platforms. Indeed, it would be preferable to reuse the same software for a wide variety of purposes.

For example, one might want to run the displays on UNIX workstations for simulation purposes. The concern here might not be so much the avionics displays themselves as the ability to monitor the performance of an aircraft simulator.

Alternatively, one might wish to run the avionics displays on desktop PCs for training purposes, for sales and marketing purposes, or even for simple communication purposes where one wants to bring a working model of the avionics display to a meeting with customers or suppliers in order to discuss the evolving specification.

As well, one would expect to run the same display software in the real embedded system installed in the flying aircraft, thereby maintaining fidelity with the approved designs and prototypes.

In short, one is looking for a system that will develop just about any avionics display to run within just about any reasonable computing environment to be used for just about any function.

Environmental Ruggedness

While the performance and flexibility of the avionics display software are key requirements, the software must also be able to run on the types of rugged COTS computing platforms that will withstand the temperature and vibration demands of a real aircraft in flight.

Increasingly, avionics designers are turning to computing systems based on the VME bus. The VME bus opens the door to the use of a range of COTS products that will provide the levels of performance and ruggedness demanded of avionics hardware, while maintaining their flexibility as to the details of the computing environment.

Finding A Better Solution

A solution to the problem of developing avionics displays must include an easy-to-use tool capable of quickly and iteratively developing complex dynamic, graphical displays with a minimum of effort or of hand-coding of software.

Once these avionics displays have been developed, it must be possible to easily and quickly enable these displays to run on a wide variety of computing platforms, again with a minimum of effort.

Overall, the solution to the avionics display development problem must accelerate the process, reduce its cost, increase its flexibility, and maintain or improve the quality of the avionics produced.

The Solution

Over the course of the last dozen years, since its founding in 1985, Virtual Prototypes Inc. (VPI) of Montreal, Quebec, Canada has been developing, improving, and extending VAPS, a tool to develop, design, prototype, test, simulate, and deploy dynamic, graphically-challenging avionics displays.

The VAPS product has matured to the point where it can meet the technical and design challenges described above. The remainder of this article describes VAPS, its application to the development of modern high-performance avionics applications and lessons learned from the process.

VAPS

VAPS provides a point-and-click interface in which users develop and test avionics displays. Having developed an avionics display in a point-and-click fashion, users can then ask VAPS to write an ANSI C program that exactly implements the avionics display they've developed.

The architecture of VAPS application programs created in this way has been carefully structured. This architecture allows the ANSI C code which implements an avionics display to be rehosted to a wide variety of COTS computing platforms using COTS cross-development tool suites with a minimum of effort.

Designing, Prototyping, Testing

One begins the process of designing avionics applications with VAPS in the VAPS Object Editor (OE). In the OE, graphics are drawn using simple point-and-click graphics design tools. Once the graphics have been designed, pieces which are to become dynamic and functional are selected and assigned functionality based on a variety of object behaviour types available within VAPS. These objects include such items as Attitude Direction Indicators (ADIs), tapes, multi-state lights, barcharts, and variable output text and numeric fields.

As well as the basic VAPS behaviour types, the user is free to combine object behaviours hierarchically so as to create even more complex behaviours.

The VAPS user has complete flexibility over the appearance of VAPS objects although a number of each type of objects are provided as starting points should the user wish to use them.

Once the user has developed the objects and graphics which are to be used within an avionics display, the user moves into the VAPS Integration Editor (IE). In the IE, the user arranges for the later manipulation of data when the application is run.

VAPS objects have plugs associated with them. Plugs are pathways into and out of an object for data. Each VAPS object type has specific plugs associated with it. For example, an ADI would have a 'roll' plug and a 'pitch' plug through which data would be fed at runtime. The appearance of the ADI would then reflect the specific data that came into it through its plugs.

The plugs from VAPS objects would typically be connected using a point-and-click interface to VAPS channels. From a user's perspective, channels are pathways for data between VAPS objects and, if appropriate, other processes. From a programming perspective, VAPS channels can be thought of as data structures. Outside processes can use VAPS channels

to pass data into a VAPS object on a display so that the object will reflect the data's value.

As well as simply passing data between VAPS objects and processes, the user can also arrange for data to be fed into or out of a VAPS Collector. A VAPS collector is a visual formalism by means of which users can define mathematical processing which is to be done on data before it is passed to VAPS output objects for the data to be displayed. Collectors are created and used in a completely point-and-click environment, so that the learning of programming is eliminated for the creation of many avionics displays.

Some avionics displays may involve more complex logic. Such a situation can arise, for example, should a portion of the display need to be changed in response to a situation where some value traverses a threshold. VAPS facilitates the creation of the behavioural logic which controls a VAPS display with the patented VAPS Stateform Editor (SE).

The SE serves as a point-and-click interface to the development of finite state machine-based logic. Users select from lists of appropriate choices state names, context-sensitive events, responses to such events, and transitions to other states. The lists of appropriate choices is filtered so that only the choices that are reasonable within the current context are presented to the user. The user has the option of adding code written in ANSI C.

The SE outputs Augmented Transition Network (ATN) programs. The ATN language is syntactically a superset of ANSI C, significantly reducing the learning curve for behavioural logic programmers already familiar with ANSI C. ATN acts to dispatch ANSI C code fragments based on events that have arisen and the logical state in which the finite state machine finds itself.

The use of ATNs as the formalism for VAPS logic permits users to select between using the point-and-click interface of the SE and using a standard text editor to create the ATN which implements the display's logic.

While VAPS provides the flexibility to include behavioural logic in the form of ATN programs, often, avionics displays require very little if any logic.

Having developed the graphics, data transfer and management instructions, and behavioural logic for a VAPS display, the display can now be tested in the VAPS Runtime Environment (RE). The RE allows the display to be animated and, if necessary, interacted with just as it will be delivered to its final user. Should there be a need to make a change to some aspect of the display, the user can return to either one or more of the OE, IE, or SE before returning to the RE to retest the display.

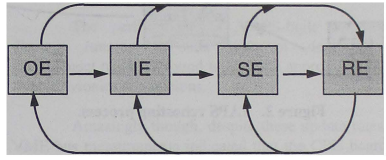


Figure 1. The VAPS Development Environment.

Rehosting

While the VAPS design environment provides the user a powerful point-and-click environment in which to develop avionics displays, that power is expanded enormously by VAPS' capability to rehost avionics displays to a wide variety of display environments.

Once the initial development of avionics displays has been accomplished, the user simply uses a VAPS command-line utility to automatically create a directory tree and to populate it with ANSI C code which exactly implements the avionics displays, their behaviours, and communications which have been designed using VAPS. As well, suitable ancillary files such as makefiles are automatically created.

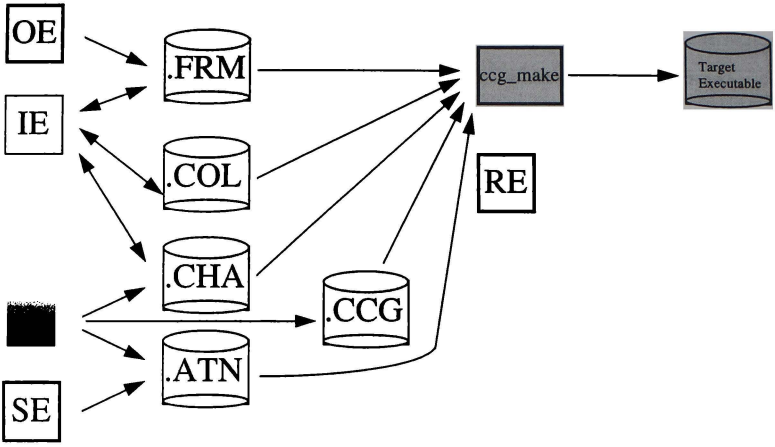


Figure 2. VAPS rehosting process.

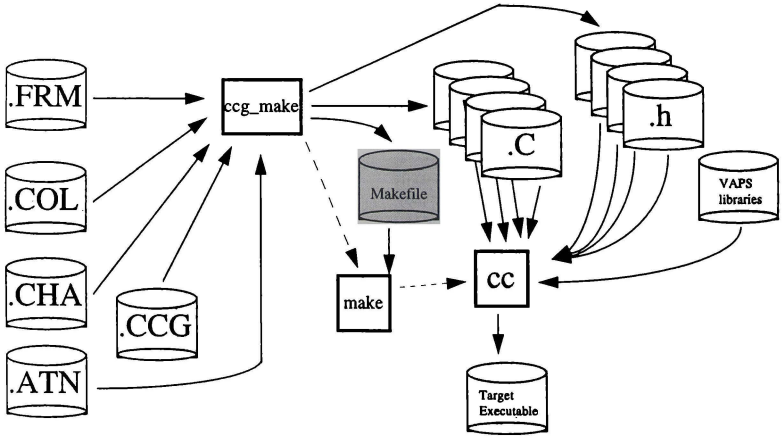


Figure 3. VAPS Rehosting process details.

Appropriate cross-compilation tools are then used to rehost the display to the selected target environment. While VAPS is perfectly capable of creating applications for UNIX- or Windows NT-based workstations, the greatest benefit of the VAPS rehosting technology is realized when VAPS displays are rehosted to ruggedized embedded systems for use as displays within glass cockpits on flying aircraft. This has the effect of completing the development cycle.

Rehosting to the Ruggedized Embedded Hardware

Avionics displays are typically deployed on such ruggedized embedded computing hardware as VME-based systems.

By porting a single file of graphics calls made by VAPS to the graphics calls supported by a specific graphics Application Programming Interface (API) VAPS avionics displays can be rehosted to an arbitrary graphics environment.

VPI has worked closely with DY 4 Systems Inc. (DY 4) of Kanata, Ontario, Canada, a leading manufacturer of ruggedized VME systems. As well as building a variety of CPU cards based on such processors as the Motorola 68K and PowerPC processors, MIPS, and Intel-based processors, DY 4 also builds graphics cards. These are built on such COTS graphics processors as the TMS34020 and the TMS340C80 (C80) graphics processors.

VPI had worked with DY 4 in order to test VAPS display applications with DY 4's Motorola 68040-based SVME 163 CPU card and DY 4's TMS34020-based SVME 770 graphics card. The VAPS graphics layer was ported to DY 4's graphical application programming interface (API) RTGS. While VAPS-built avionics displays ran successfully in this environment and several customers have found a use for this environment, avionics displays on this environment tended to run too slowly to be useful.

DY 4 recently developed the PowerPC 603-based SVME 170 CPU card and the C80-based SVME 783 graphics card. The performance of the SVME 783 far exceeds that of the SVME 770 owing to DY 4's clever implementation of its RTGS API on the SVME 783. VPI has worked closely with DY 4 to enable VAPS-built avionics displays to be rehosted to RTGS for the SVME 783 graphics card.

In this environment, the VAPS display process runs on a VxWorks RTOS on the SVME 170 CPU card, making such graphics calls as line, rectangle, polygon, and attribute-specification requests through the RTGS API. The RTGS functions, in turn, translate high-level API functions into token-parameter tuples which are passed across the VME bus to the SVME 783 graphics board where the tokens and parameters are read by the RTGS command processor, RTGS software which runs on the SVME 783's C80 processor. The C80 then draws the graphics whose requests were represented by the token-parameter tuples retrieved from the VME bus.

Performance

The performance of VAPS-built avionics displays running in this type of deployment environment has been found to be quite appropriate for flyable avionics applications.

Amazingly though, despite these update rates, VME bus measurements indicated that the CPU board was spending a lot of time doing nothing more than checking to see whether or not the graphics board had finished the work that had been assigned to it. Typically this would have involved a single update of the display. Unfortunately, the process of checking on the graphics board's progress consumes both VME bus bandwidth and CPU resources.

In real avionics applications, VME bus bandwidth may well be required for the internal transmission of other data such as, for example, sensor data. It seemed wasteful to consume VME bus bandwidth simply to check on the graphics board's progress.

Rehosting directly to the 783

As was discussed above, the VAPS rehosting process involves the generation of ANSI C code. This approach greatly enhances the portability of VAPS-built applications. With DY 4's help, VPI took advantage of this VAPS architecture and the fact that RTGS provides the same API on the SVME 783 as it

does on the SVME 170 to rehost VAPS displays directly to the C80 processor on the SVME 783.

This was largely a matter of simply recompiling the VAPS libraries for the C80 processor with the Texas Instruments cross-compilation tools although minor adjustments had to be made because of the way in which the C80 handles caching. Having ported the libraries, however, any VAPS application could now be cross-compiled to run on the SVME 783 directly, without the intervention of the SVME 170.

What is done in practice is to turn the executable program for the SVME 783 into a large data array for a program on the SVME 170. The SVME 170 transmits the data array as a downloadable function to the SVME 783 at initialization time, then sets it running.

From then on, the VAPS-built display runs entirely on the SVME 783, needing only application data to be passed into it. It is no longer dependent on the SVME 170 for the execution of graphics commands.

Problems solved

The VME bandwidth that would otherwise have been consumed by having the SVME 170 constantly sending graphics commands and then checking on the SVME 783 is no longer consumed during steady-state operation of the display and is available for use either by the remainder of the system or as a safety buffer. Similarly, after initialization, CPU resources are no longer consumed for the purpose of updating avionics displays.

As well, customers now find themselves in the advantageous position of using a single SVME 783 graphics card per display, rather than having to use a combination of a graphics card and a CPU card for each display.

Customers report typical graphical refresh rates between thirty and sixty hertz (30-60 HZ) on real, double-buffered avionics displays deployed directly on the SVME 783 and intended for use in flying aircraft.

The net effect of the improvements afforded by the rapid rehosting of VAPS-built avionics displays is that avionics displays can be made to run fast and smoothly enough to serve as real avionics displays in flight while reducing hardware costs and system resource consumption.

The VAPS tools themselves significantly reduce the level of effort required to build the avionics displays while their quality is enhanced due to the short, iterative, and intuitive nature of the design cycle.

Documentation

While VAPS focuses on rapidly building efficient, deployable, high-performance avionics displays, it also provides tools to assist avionics developers in documenting the avionics displays they create with VAPS.

Users of the VAPS design environment can arrange to have VAPS output an ASCII text description of the displays they have created in the form of a VAPS Graphics Metafile (VGM, or metafile, for short). VAPS translates readily and rapidly between the graphical file format it normally uses (VAPS Frames) and VAPS metafile. The metafile can be stored off-line with configuration control information or included in textual documents as a formal description of the avionics display in question.

Within the VAPS OE, users can optionally record textual comments in relation to each VAPS object that is created. When used, these comments become part of the definition of the object in question.

As well, VAPS-built displays can be printed or output as postscript files for inclusion in specification, training, or other user documentation.

The VAPS/Help Engine allows users to link HTML files directly to a VAPS-created object. The link between the object and the HTML file becomes part of the definition of the object itself. The HTML files in question can be used either as interactive job-aids, training materials, or even as on-line specification or revision control documents.

As well as providing textual and printed design documentation, the ANSI C code which VAPS outputs is thoroughly commented. Each element of the data structures produced is documented as is each function. As well, each ANSI C function is commented in the generated code so as to link it directly to the VAPS component which it implements.

A solution to COTS requirements.

The VAPS product provides a completely COTS solution to the development of displays for avionics. The VAPS display environment runs on common COTS workstation environments and can as easily target COTS and proprietary computing, graphics, and RTOS environments.

Effectively, VAPS and its unique rehosting capability allow users to develop high quality avionics displays while minimizing concerns about the details of the deployment hardware and RTOS. As long as the RTOS and computing environment provide the necessary facilities, and the graphics environment provides the necessary graphics infrastructure, a VAPS-built display can be made to run in that environment.

Graphical API

The use of the X Consortium's X11 graphical API has emerged as a frequent request with respect to the development of avionics displays over the last couple of years. The most frequently cited reason for using the X11 API is to ensure the long-term portability of the system.

In our experience, regardless of whether the computing platform is a general purpose workstation or an embedded system, the use of the X11 graphical API always has the effect of reducing the graphical refresh rates of relatively demanding avionics displays.

The use of the X11 graphical API to enhance the portability of avionics applications seems particularly counter-productive. Not only does the use of the X11 API slow down the updating of avionics displays, but the complex X11 API itself must first be ported to the proposed hardware. Combined, the requirement to provide higher-end hardware to support the X11 API, and the cost of porting and tweaking the X11 API to the hardware in question far exceeds the cost of using a tool like VAPS. While VAPS can rehost to X11 environments, it can easily rehost higher-performing displays to the graphics API native to the specific graphics card in question, typically in a shorter over-all time frame for a project.

Specifiers of avionics systems should consider the benefits of using VAPS to provide a higher-level solution to the portability problem than is allowed simply by using the X11 API. When project requirements are prepared, manufacturers should be allowed the necessary flexibility to assure application portability through other means than the use of the X11 API.

The Future

VAPS now provides an increasing number of avionics manufacturers significant productivity and quality gains while reducing their avionics display development costs and reducing their products' time-to-market

DY 4's SVME 783 graphics card is capable of merging live video with such synthetically generated graphics as those resulting from the execution of an avionics display built with VAPS. Both an analog video input capability and a digital one are available with the optional mezzanine card which will soon be available for the SVME 783. VPI and DY 4 are looking forward to demonstrating VAPS applications running in concert with live video import when the mezzanine card becomes available for the SVME 783.

As well, there may still be opportunities to improve the cost benefits of using VAPS. The DY 4 SVME 783 can, with the mezzanine card mentionned above, output simultaneously to two graphics display devices. If this capability can be successfully harnessed to VAPS displays, users may be able to further reduce the number of graphics cards required to implement a multi-headed avionics suite.

Conclusion

VAPS is a COTS tool which facilitates the development and deployment of avionics displays on a variety of COTS computing and display environments.

These include the ruggedized VME environments on which avionics applications are typically deployed.

Significant productivity, quality, portability, and time-to-market benefits can be obtained by using VAPS with such COTS products as VxWorks running on DY 4's SVME 170 and SVME 783 computing and graphics cards.

While success has been described with specific DY 4 products, the inherently open nature of the VAPS rehosting process allows users to select the specific computing and graphics products they will use so long as they provide the resources and services necessary to the task at hand.

COMPOSITE MATERIALS TECHNOLOGIES REDUCE COST OF HIGH-DOLLAR COMPONENTS FOR MAINTENANCE TRAINING

JAMES P. EXTER
METTERS INDUSTRIES, INC.
ORLANDO, FLORIDA

ABSTRACT

New composite, urethane, polymer and fiberglass materials have made the simulation of high-dollar mechanical and Line Replaceable Unit (LRU) components possible for maintenance training device applications at a substantially reduced cost. Utilizing much lower cost materials and processes, complex mechanical systems such as engines and transmissions for fixed wing, helicopter and ground vehicle systems can now be built for a fraction of the real aircraft systems cost, thereby freeing up real components to go back into the fleet. LRU's can be molded using low-cost materials with a high degree of fidelity including functioning connectors, hardened attachment points, steel, springs and pivot points where appropriate and the necessary weighting and color. This technology reduces pressure on a military supply system which is clearly under funded for the weapon system and the associated maintenance training devices. The process for choosing the best materials taking into consideration the systems' operational requirements such as operational needs for training i.e. remove and replacement tasks, electrical trouble shooting, torque specifications, hardened attachment points, fluids changes and system inspections is presented in this paper. There is also a discussion of the range of materials currently being utilized in developing simulated vehicle components and the decision variables which should be considered when selecting materials.

STATEMENT OF NEED

New maintenance training approaches are required to teach weapon systems maintainers job skills if proficiency levels are to be maintained. The number of new weapon systems starts is clearly in decline as the United States moves from a relatively high single threat environment requiring large numbers of diverse weapons systems into today's multi-threat environment requiring fewer more specialized high technology weapons systems. With the decline of the number of new weapon systems starts comes pressure on existing weapons systems to continue in service longer. The unit and subsystems costs increase with weapon systems complexity putting even more pressure on the limited weapons systems budget available. Fewer weapons systems, higher unit costs, increasing system complexity and fewer numbers of soldiers/maintainers have all put extreme pressure on the current maintenance training infrastructure. Student throughput is also an issue forcing pressure on the existing maintenance training systems as training hours go up to cover increasingly complex systems/architectures.

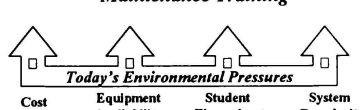


Figure1. Today's maintenance training environment.

It should also be noted spare parts are in great demand due to the extended life of today's weapons systems. Spare parts once available for maintenance training are now needed to keep weapons systems operational.

All of these factors contribute to the need for improving our maintenance training approaches and tools.

TRAINER MATERIALS SELECTION PROCESS

The need to change thinking when developing weapons systems maintenance training devices is upon us if we are to keep pace with military training demands in today's environment. Pulling components off the weapons systems production line is no longer acceptable considering unit costs, weapons systems priorities and limited availability. Considering all the above reasons for not utilizing actual weapon systems subassemblies, other ways of building

Copyright © 1997 by the American Institute of Aeronautics and Astronautics, Inc. All rights reserved.

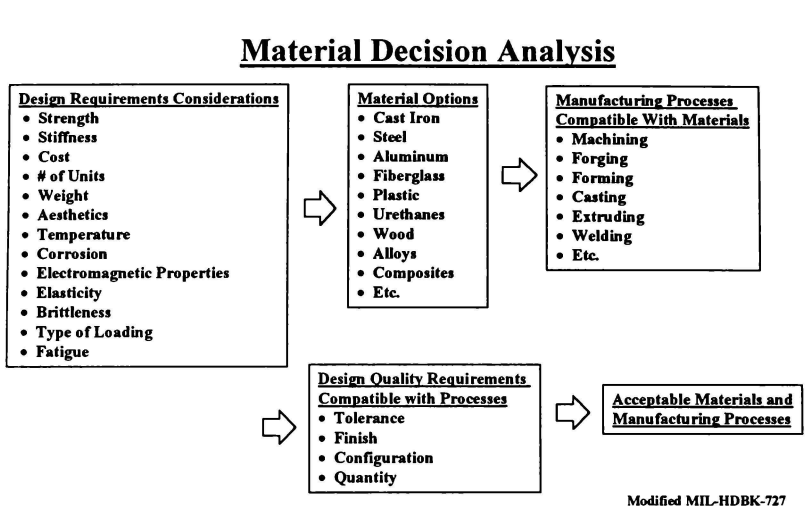


Figure 2. Material Selection Process

maintenance training devices must be identified. Figure 2, Material Decision Analysis, allows the design engineer to quickly review important criteria for building maintenance training components. When choosing materials for maintenance training components it is almost assured the materials used for the real components will be cost prohibitive. Additionally, the environmental conditions that real systems are subjected to go far beyond maintenance training requirements. Clearly the effects of war are not part of the maintenance training environment.

Often plastics, fiberglass and polymers can be substituted for milled aluminum or machined steel components. For maintenance training components the use of composite materials can support a range of applications from space claim items, to functioning Line Replaceable Units (LRU), to complex mechanical systems like the AH-64 Apache Helicopter's transmission. Composite materials are formed or cast by using both soft and hard tool molds. Molding processes and materials must be chosen together depending on part requirements. Hard tool molds are utilized if heat curing is required or the material chosen is fiberglass or composite woven material requiring mold stiffness and strength where parts must be formed. When the material chosen is a liquid such as a two part polymer or resin and can be poured, often soft tooling is used. Hard tools are made with steel, aluminum or special heat resistant fiberglass. Soft tools are made with silicon based or rubber material. Plaster or plastic-faces plaster can also be used for building molds if there is only a limited number of part reproductions required. Figure 3. describes many of the mold building materials and their characteristics.

Although part materials must be chosen prior to selecting the molding/tooling material and process, a number of part materials can be used with each type of tooling. Part materials are chosen depending on their application but some considerations are strength, stiffness, weight, temperature, corrosion, type of loading, cost and electromagnetic properties. Material options for molded products are plastics, fiberglass, urethanes, and composites. It should be noted the term composite materials refers to a range

of urethanes, fiberglass or two-part materials. Part production is also a major consideration when choosing materials as mold/mold materials often have a finite life. This is particularly true when heat curing is required or when extreme heat is generated during the curing process.

Mold Making Material

Material	Characteristics
Silicone	Fine Detail and Sharpness No Mold Release Expensive Limited Production (20-50 units)
Plaster	Reasonable Detail Requires Mold Release Inexpensive 1 or 2 Reproductions
Plastic-Faced Plaster	Reasonable Detail Requires Mold Release Relatively Inexpensive Limited Production (6-10 units)
Urathane	Fine Detail (not compatible w/ all Materials) No Mold Release Expensive (Less than Silicone) Limited Production (10-30 units)
Fiberglass ■ Polyester ■ Epoxy	Good Detail Requires Mold Release Very Expensive High Production

Figure 4. Mold Making Material Options

Two part molds are often required to maintain the necessary tolerances. Two part/multi-part molds essentially force the part material to take the shape of the mold. Another approach commonly used is a vacuum bagging approach whereby the vacuum bag applies pressure to the part material forcing the material to take the desired shape.

Part Material

Material	Cure Time	Comments
Pre-Preg (frozen Material)	4 Hours	Very Clean Precision Material Light and Strong Expensive
Fiberglass	2-3 Hours	Very Durable Gel Coat or Paint Many Applications Relatively Inexpensive
Urethanes (2 part) 1. Rigid 2. Semi Rigid 3. Elastomer 4. Flexible	15 min - 24 Hours	1. Hard but Brittle 2. Hard but Brittle 3. Strong Durable Flexibl 4. Durable, Rubber Texture
Urthane Foam	15 Min	Used for Filling and Flotation
Latex Foam	8 Hours	Cushion Material arms rests, etc.

Figure 5. Part Material Options

Mold making material and parts material manufacturers /suppliers can provide the necessary work process information. In fact, it is often necessary to follow the prescribed manufacturers process if the manufacturer is to warranty the materials.

COMPOSITE MATERIAL EXAMPLES

Two examples of the concept of the composite material for simulator development applications are as follows. The first example is the AH-64 Apache Transmission which illustrates the value of using composite materials for simulated aircraft mechanical systems maintenance training requirements. Cost savings for using composite molded parts for mechanical systems applications is around 75% considering the Apache Transition example.

The second example of the use of composite molded materials technology is the M1A2 Main Battle Tank Line Replaceable Unit (LRU) components for the M1A2 Maintenance Trainers. Composite material LRUs are perfect for maintenance training applications as they are actually often more durable then real components and therefore have a much longer life. Composite materials can also be used for a number of other ground vehicle systems including main guns, hydraulic fluid tanks, lighting and intercom boxes, and virtually all attached crew compartment components.

AH-64 APACHE TRANSMISSION

In 1994 Metters Industries signed a cooperative research and development agreement (CRADA) with Oak Ridge Laboratory to develop a composite version of the Apache Transmission. The notion was to put the rather scarce real aircraft transmissions back into fleet supply while utilizing a composite version for the maintenance training application. During 1995, the real transmission was disassembled and for each of the components, mostly gears, a mold was constructed so the components could be duplicated. Molds are built by suspending the part to be molded within a wooden or plaster structure so the mold material can flow around the part (Figure 6). The molds wooden structure is made in two parts so when the mold material has cured the part can be easily removed. The part to be

molded is often coated with mold release to facilitate part removal.

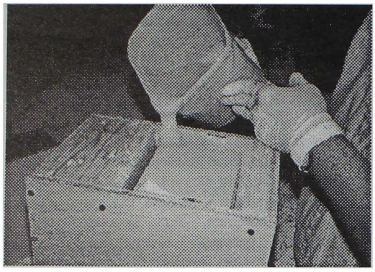


Figure 6. Mold Building Technique

For most of the gearing, a two part urethane elastomer was chosen to provide the necessary strength, durability and elasticity necessary. Once the mold is constructed, the part material is poured into the bottom part of the mold and the top of the mold is put into place (Figure 7). There are specially constructed holes in the top of the mold to allow the part material to be poured into the mold to complete the part. The mold, with the part material inside, can be put into a pressure chamber to help reduce air bubbles sometimes a problem with certain types of parts materials. When parts are removed from the mold they can be easily finished and painted as required and prepared for assembly.

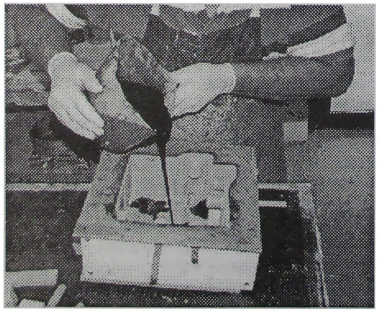


Figure 7. Part Building Technique

The Apache Transmission outer shell was built using a fiberglass material except for the top piece. The top piece of the transmission case is also the attachment point for the unit to the fuselage frame structure. The top piece of the transmission was built using spun wound composite material to provide for the extra strength required to support the weight of the transmission.

After completing the duplication of the transmission components, additional steps had to be taken to insure proper system operation. Bearings were machined out of aluminum and steel inserts were set into the transmission housing so torque specification can be maintained. The composite transmission has been tested at speeds up to 20 RPM without a rotor load just to insure system operation for maintenance training tasks requiring system operation.

M1A2 MAINTENANCE TRAINER LRUs

After reviewing the availability and cost of using actual M1A2 Line Replaceable Unit components for the M1A2 Maintenance Trainer, it became obvious a different approach was necessary to meet schedule and cost constraints. Although many of the actual tank components were available on loan for mold building purposes, using the real LRUs was determined to be cost prohibitive. In most instances, the LRUs need only to be weighted correctly and have the proper mechanical and electrical interfaces/ connections. Figure 8 below shows a mechanical latch assembly made from a molded urethane material with machined metal parts were required. Note the part on the right is the supplied GFE while the part on the left is the simulated component.

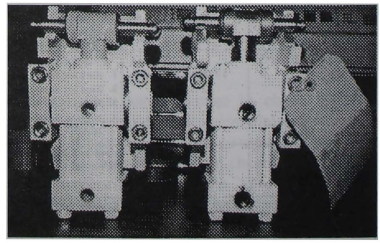


Figure 8. Composite Material Latching Assembly

The cost savings associated with the composite molded component approach for LRUs is much more difficult to determine as the individual LRU costs vary. However, from 5 to 20 percent is a good rule of thumb.

SUMMARY

Today's peace-time defense environment is limiting the number of new weapon systems starts and therefore forcing existing weapons systems platforms to be upgraded and kept in service long beyond their original design life. This also put significant pressure on maintenance training devices as many of the components are no longer in stock or being produced by the original equipment manufacturer. The availability of low cost substitute items will increase the stock of complex mechanical components for real and simulated applications. This will help reduce the pressure on the supply system and support the weapons systems fleet by increasing readiness with greater availability of critical mechanical/LRU components while reducing training systems costs.

AIAA-97-3802

Visual-Vestibular Interactions in the Perception and Control of Aircraft Motions by the Pilot

Ruud J.A.W. Hosman*
Delft University of Technology, Faculty of Aerospace Engineering, Kluyverweg 1, Delft, the Netherlands. E-mail: *r.hosman@lr.tudelft.nl*

Abstract

At the Delft University of Technology, a long-term research project on the perception of aircraft motions by the pilot is in progress. The most recent results of this research show that the contributions of the central and peripheral visual system, the vestibular system, and the interaction between these systems are of primary importance to the perception of aircraft motions. Considering the results obtained so far, a simple linear model, incorporating the visual and vestibular sensors, has been developed. With this model, the influence of the visual and vestibular motion cues, the simulation time delays, and the washout filters on the pilot-simulated aircraft closed loop can be analyzed. The results show that the present maximum allowed time delay of 150 ms has a stronger influence on pilot's performance and required adaptation then the classical washout filters.

1. Introduction

Due to economical and environmental pressures in civil aviation in Europe, there is a growing impetus to increase the use of simulators for pilot training. This aim has already been achieved for pilot type-conversion and type-recurrent training. For ab initio pilot training, on the contrary, simulation technology still has a long way to go. Extension of the application of flight simulation to the skills development of young pilot's exclusively in synthetic environments, however, requires a full understanding of a pilot's perception and control behavior in manual aircraft control.

To improve the design and control of the simulator motion cueing systems, a better understanding of the contribution of motion cues is required. During the manual control of aircraft, the perception of the vehicular motions is primarily based on the pilot's visual and vestibular perception. Due to the simulation hardware there are, however, a number of differences between the information provided in the real and the simulated vehicle. The most important of these are:

- The equations of motion of the simulated aircraft are solved by a digital computer with limited sample frequency and inherent time delay.
- The outside world is simulated by displaying a computer-generated image with limited update frequency and a finite computational time delay.
- Due to the physical limitations of the motion system, the aircraft motions have to be transformed to the motions of the simulator motion-system by washout filters with their inherent dynamic characteristics[11].
- The mass properties of the simulator moving platform can induce phase lags in the response of the motion base[1].

These limitations can cause a delayed response in the visual and the vestibular cues, and there may be a noticeable mismatch between these signals. The phase characteristics of the motion washout may introduce a significant penalty to the temporal fidelity. Furthermore, the washout filters are mostly optimized based on minimization of the error in vestibular motion perception. In the literature, these two problems are normally treated separately.

From the results of research at the Faculty of Aerospace Engineering of Delft University of Technology during the seventies, it became clear that the vestibular sensory output alone was not the right basis to optimize simulator motion system control [4]. Since the ultimate goal of flight simulation is that pilot's perception of aircraft motions in simulated flight are equal to those in real flight, this can only be reached by optimizing the simulator motion system control under the condition that the perceived aircraft motions, based on the visual and vestibular cues in real and simulated flight, are equal. Such an optimization is only possible when the characteristics of the visual-vestibular motion perception process are described in adequate detail.

Based on these considerations, it was decided to start an investigation on the contributions of the visual and the vestibular system to the perception process and the consequences for a pilot's control behavior. For this study a comparison of the specific characteristics

*Associate professor, Faculty of Aerospace Engineering

Copyright © American Institute of Aeronautics and Astronautics, 1997. All rights reserved.

of the visual and the vestibular system in motion perception was necessary.

The tracking task was considered to be too complicated to determine the differences between the visual and vestibular contribution to the perception process. It was decided therefore to investigate the visual and vestibular perception of motion by using stimulus-response tasks with well-chosen static and dynamic stimuli, and to evaluate the results with a mathematical model of control behavior.

Based on the literature on human information processing, the human operator has to be considered a single-channel information processor with limited capacity, and with multi-sensory input [16]. If that model is correct, then there is no reason that it can not be applied to the closed-loop control task. Using this model, the stimulus-response tasks with discrete and dynamic stimuli were designed to investigate the contribution of the central or foveal visual and peripheral visual system on the perception of the aircraft roll angle and roll rate. In addition, dynamic stimuli were applied to establish the differences between the speed and accuracy of motion perception with the visual and/or the vestibular system. Beside these stimulus response experiments, a tracking task experiment was performed to obtain a data base on the influence of visual and/or vestibular motion feedback on the pilot's tracking performance and control behavior.

The results of the stimulus-response experiments, together with the well-known characteristics of the visual and the vestibular system, provided the data to set up a descriptive model [3,6,7]. The validity of this model was evaluated with the frequency response data from the tracking task data base [5,7]. Finally, the experimental results and the descriptive model provided a solid knowledge of the influence of motion feedback on pilot's control behavior [7].

The proposed descriptive model opens the opportunity to investigate the influence of visual and vestibular motion cues, time delays, and washout filters on the open and closed loop characteristics of the pilot simulated-aircraft dynamics.

Normally, the pilot experiences small differences between the simulated aircraft and real aircraft characteristics. Due to the high adaptability of the human operator, the pilot will adjust his control behavior in such a way that he will reach optimal performance and adequate stability with the simulated aircraft. It is exactly this sub-conscious adaptive compensation in the simulator which may influence the perception of the vehicle characteristics or handling qualities. For a maximum transferability of training, this adaptive compensation by the pilot should be understood and, where possible, reduced. If the descriptive model is accepted as an adequate description of the information processing by the pilot, then the required adaptation of the model to optimize the performance may be considered as an indication of the adaptation of the pilot.

In this paper, the descriptive pilot model will be discussed. Thereafter, the model is used to analyze the influence of the pure time delays and distortions caused by the washout filters on the pilot's performance and adaptation in the simulated aircraft pitch control task.

2. Model background

Three groups of experiments, designed and executed to investigate the perception of aircraft motion and its influence on pilot's control behavior and performance, have been described [6,7]. Two of these experiment groups were performed to establish the accuracy and speed of the visual and vestibular perception of aircraft attitude and angular rate. In these trials, static and dynamic input stimuli were used in stimulus-response tasks. The third and final group of tests was performed in tracking tasks with different combinations of central visual (artificial horizon), peripheral visual (outside visual display), and vestibular or whole-body motion cues. To develop a model for the perception and control of aircraft motions, the results of the individual experiments were placed into one framework. The basis for this framework was the model describing the information processing [16], Fig. 1.

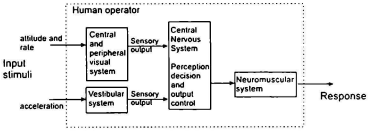


Figure 1. The visual- and vestibular pathway of information processing in the control task.

It was argued that more than one sensor may be involved in the perception of a certain motion stimulus [7]. In the case of aircraft motion perception, the visual system, the vestibular system, and the proprioceptive system are all sensitive to motion inputs. It was shown that the visual and the vestibular system dominate the motion perception process and that an adequate description of that process can be obtained based on these systems. The input stimuli to the sensors are processed in parallel and transformed to sensory output, while the sensory outputs are the input to the perception process in the central nervous system. The subsystems of the model are the visual and vestibular sensors, the perception and decision processes in the central nervous system, and the neuromuscular system generating the control output.

From a survey of literature [8,15] and research at the Delft University of Technology [5], it is known that vestibular and peripheral visual motion feedback has a beneficial effect on pilot's control behavior and performance. In the perception and decision process, the sensory outputs are transformed to pilot's control output. When pilot's control behavior is considered as skill-based behavior [13], which was optimized during the extensive training of the pilot, it may then be assumed that the control output results from a weighted sum of the sensory outputs.

Considering the characteristics of the visual and the vestibular system [6,7], the dynamic characteristics of the relevant visual detection systems and the vestibular system could be determined. It became clear that a distinction has to be made between the visual perception of attitude, and of angular rate in the central visual field. Similarly, a distinction has to be made between the perception of angular rate in the central, and that in the peripheral visual field. The relevant conclusions based on the results of the attitude and angular rate perception experiments are:

1. The reaction time for angular rate perception with the central visual field from the artificial horizon is longer (100 to 150 msec) than for attitude perception.
2. The reaction time for angular rate perception from the peripheral visual field is shorter (about 60 msec) than that from the central visual field.
3. The perception of roll rate from the peripheral visual field and that from the central visual field are different processes.

Using the experimental results, the relations between the inputs to the visual system and the visual sensor output can now be described. These relations, which are described by time delays, are considered to be fixed relative to each other. The perception of the aircraft attitude in the Central visual field can be described by:

$$H_{C,att}(\omega) = Ke^{-j\omega\tau^*}, \qquad (2.1)$$

for perceiving attitude *rate* in the Central visual field:

$$H_{C,rate}(\omega) = Ke^{-j\omega(\tau^*+\tau_C)}, \qquad (2.2)$$

and for perceiving attitude *rate* in the Peripheral visual field:

$$H_{P,rate}(\omega) = Ke^{-j\omega(\tau^*+\tau_P)}. \qquad (2.3)$$

The delays τ^*, τ_C and τ_P were defined by the experimental results obtained: τ^* = 50 msec., τ_C = 100 msec and τ_P = 40 msec. Thus the output of the central and peripheral-visual system due to attitude and rate stimuli can be described as shown in the upper part of Fig. 2.

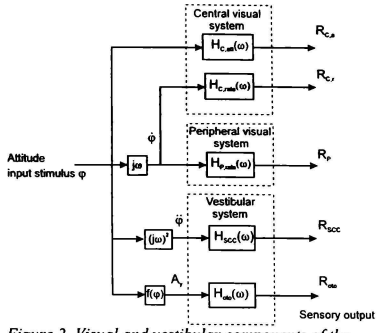


Figure 2. Visual and vestibular components of the sensor system.

Three stimulus-response experiments to investigate the interaction between visually and vestibularly motion cues were also conducted, in which a second-order system step response was used as the stimulus [6,7]. The main conclusions, based on the results, were:

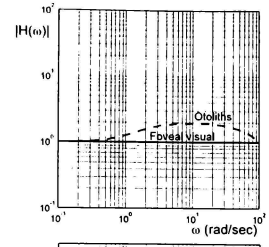
1. The accuracy of the estimation of the final value of a complex motion stimulus is practically independent of the display configuration, visual stimulation, vestibular stimulation or all of these.
2. The reaction time to the step-magnitude estimation is significantly decreased (200 to 400 msec) by adding cockpit motion to the visual presentation of the stimulus to the subject.
3. In motion perception, cockpit-motion cues, as well as peripheral-visual cues, speed up the perception process.

The faster vestibular perception of motion, relative to the visual perception, results from the dynamic characteristics of the vestibular system, on the one hand, and of the longer neural processing time of the visual system, on the other. Considering the results, the scheme of Fig. 2 was extended with the vestibular system in the lower part of the figure.

The dynamic characteristics of the semi-circular canals and the otoliths are described by transfer functions [3]. The characteristics of the otoliths have to be incorporated into the model if the response contributes to the perception process of the controlled variable. The transfer functions of the semi-circular canals and the otoliths are:

$$H_{SCC}(\omega) = \frac{K_{SCC}(1+\tau_n j\omega)}{(1+j\omega\tau_1)(1+j\omega\tau_2)}, \qquad (2.4)$$

and



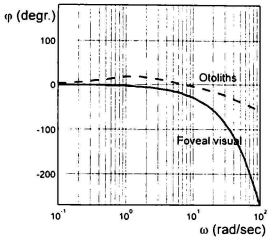
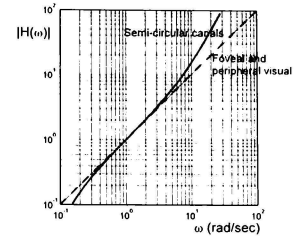


Figure 3a. The transfer functions of the attitude sensitive sensors, the central visual system and the otoliths.



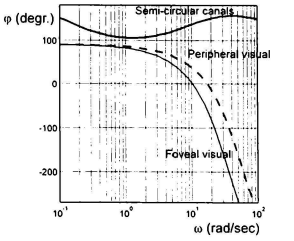


Figure 3b. The transfer functions of the rate sensitive sensors, the central and peripheral visual system end the semi-circular canals.

$$H_{oto}(\omega) = \frac{K_{oto}(1+\tau_n j\omega)}{(1+\tau_1 j\omega)(1+\tau_2 j\omega)} \qquad (2.5)$$

In Fig. 3, the transfer functions, describing the relation between the roll attitude input with sensory output, are presented. Fig. 3a presents the Bode plots for attitude output of the central-visual system (Eq. 2.1) and the otoliths (Eq.2.5). Note that the otoliths' output leads the central visual system output. Fig. 3b shows the Bode plots for the rate-sensitive sensors in the central- and peripheral visual system and the semi-circular canals. The transfer function of the semi-circular canals, (Eq. 2.4) relates input angular acceleration with the neural sensory output in impulses per second. To make the transfer function of the semi-circular canals comparable to that of the visual system, the input had to be changed to roll attitude and the gain had to be adjusted so that the same modulus is obtained for visual rate perception at $\omega = 1$ rad/sec. The semi-circular canals output leads to the visual system output.

The well-trained and experienced pilot has developed an efficient control strategy using the available sensory information to generate his control output. From tracking experiments with different combinations of central visual, peripheral visual and vestibular feedback of motion information, it has been shown that a weighted sum of the sensory input is the most likely input to the single-channel information process [7], as depicted in Fig. 1. In Fig. 4, the knowledge of the sensor dynamic characteristics and the control behavior is summarized in one descriptive model. The perception and decision process starts with the weighting of the sensory outputs by a weighting factor K for each individual sensor, $K_{C,att}$, $K_{C,rate}$, $K_{P,rate}$, etc.

It is assumed that the processing in the perception and decision stages by the Central Nervous System can be described by a gain K_I and a lumped time delay τ. This time delay is considered to be dependent on the task to be performed, but independent of the display configuration. For the stimulus-response task, for instance, the reaction time is, according to the Hick-Heyman law, dependent on the information content of the stimulus. In the case of tracking tasks the time delay τ may also be dependent on the specific tracking task, disturbance task or target following (maneuvering) task.

The information processing is finally completed by the neuro-muscular system to generate the control output [10,12]. The neuro-muscular-manipulator system of the well-trained human operator can be approximated over a wide frequency range by a third order transfer function. For the investigation of the influence of motion perception on the pilot's control behavior, the bandwidth of the neuro-muscular

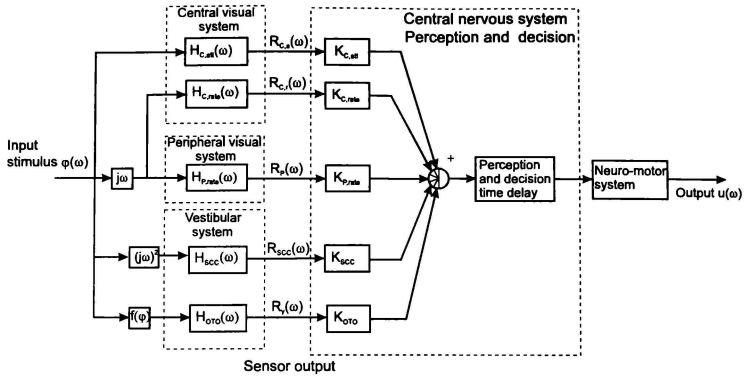


Figure 4. Block diagram of the descriptive model for the information processing by the human operator controlling a vehicle.

dynamics is high and relatively unimportant. Therefore, a pure time delay, τ_{NM}, can be used as a low-frequency approximation, up to ± 10 rad/sec, of the contribution of the neuro-muscular system to the subject's control behavior.

For further application of the descriptive pilot model, the time delays, resulting from the information processing in the perception and decision stages and the neuro-muscular system, are combined and described by one time delay for the entire information processing:

$$H_I(\omega) = K_I e^{-j\omega\tau_I}, \qquad (2.6)$$

with $K_I = 1$.

The application of this descriptive model to the pilot's control behavior in aircraft control and flight simulation has two specific advantages:

1. Due to the limited number of unknown parameters, the weighting factors K, the model can be easily adapted to a specific control task.
2. Due to the incorporation of the individual sensors in the model, the influence of the specific simulation characteristics, time delays and washout filters, on the pilot's control behavior can be investigated.

To explain the analysis in the next session, the block diagram of the model in Fig. 4 is simplified. In Fig. 5, a scheme of the model is shown in the closed-control loop. In this scheme the relation between the model attitude stimulus and the weighted sensory output is replaced by one block for the central visual system, and another block for the feedback of motion cues; peripheral visual cues, vestibular cues or both. The aircraft dynamics are described by the block $H_c(\omega)$.

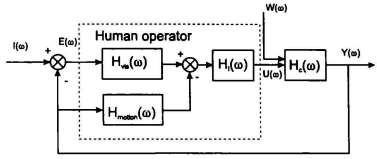


Figure 5. Simplified scheme of the descriptive pilot model in the control loop.

3. Analysis

Due to the high order of the aircraft dynamics, in manual aircraft control the pilot's control task is organized in nested control loops, Fig. 6. The bandwidth and stability of the inner (pitch, roll, and yaw) attitude loops determine the control characteristics of the aircraft. The pilot pays his attention first to the control of the aircraft attitude. Therefore, it is understandable why in flight simulation specifications a preference for the proper simulation of the accelerations in the appropriate rotational axis is recommended [2]. Therefore, the analysis will concentrate on the simulation of the rotations.

There are two factors influencing the aircraft inner control loop. One is the reference attitude for the inner loop when maneuvering the aircraft. The tracking task to follow this reference will be called the target following task, sometimes called "maneuver motion" [9,14]. The other source is atmospheric

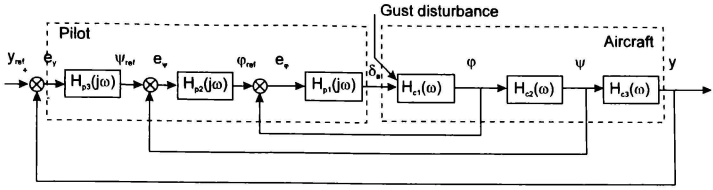
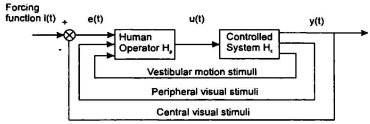
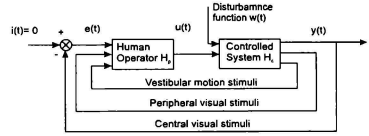


Figure 6. Nested control loops for the lateral aircraft control during an approach to land.

turbulence disturbing the aircraft attitude. Minimization of the aircraft attitude response by the pilot to external disturbances is called the disturbance task or "disturbance motion" [9,14]. The impact of these two factors on the inner control loop and consequently on pilot's control behavior is fundamentally different. This difference is enhanced when aircraft motions are fed back to the pilot by the outside world stimulating the peripheral visual field and/or by the vestibular system, Fig. 7.



a. Target following task



b. Disturbance task

Figure 7. Feedback of peripheral visual and motion cues in the target-following in the disturbance task.

In the target-following task, the attitude error, presented on the primary flight display, is minimized by the pilot. As shown in Fig. 7 the peripheral visual and vestibular motion cues are not correlated with the error attitude signal on the PFD. This is also shown in the simplified scheme of the model in Fig. 5. As already mentioned, in this scheme the relation between the model attitude input and the sum of the weighted
sensory output is replaced by one block for the central visual system and one block for the motion perceiving sensor, the peripheral visual system, the vestibular system or both. Hence, for the target following task the pilot transfer function is:

$$H_{p,Foll}(\omega) = \frac{H_{vis}(\omega).H_I(\omega)}{\{1 + H_{motion}(\omega).H_I(\omega).H_c(\omega)\}}. \tag{3.1}$$

In the disturbance task, however, the feedback motion signal is correlated with the presented attitude on the PFD. In that case, the pilot transfer function is:

$$H_{p,Dist}(\omega) = \{H_{vis}(\omega) + H_{motion}(\omega)\}.H_I(\omega). \tag{3.2}$$

In Fig. 8 the bode plots of the transfer functions. describing pilots control behavior in the target following task (maneuver motion), and the disturbance task (disturbance motion), with and without motion feedback are presented. Clearly, the influence of motion feedback is different in the two tasks. In the target following task, the pilot is able to increase the stability of his control, generates accurate control input to the vehicle and improve tracking performance. In the disturbance task, the pilot is able to increase the bandwidth of his control and improves the tracking performance.

When the simulation time delay τ_s and the washout filters, $H_{MF}(\omega)$, are incorporated in the closed-control loop with the pilot and the simulated aircraft, than the simplified scheme of Fig. 5 changes to the scheme of Fig. 9. Consequently, the characteristics of the open loop, between the error signal e and the aircraft output y, changes. These changes cause a decrease in control-loop bandwidth, a change in stability, a decrease in tracking performance, and forces the pilot to adapt his control behavior. In control of the simulated aircraft, the pilot tries to maintain the standard pilot-aircraft behavior. Therefore, the transfer function describing the pilot's control behavior changes. These changes result from the time delays and the washout filters, on one hand, and from the adaptation by the pilot on the other. For the target following task the open-loop transfer function becomes:

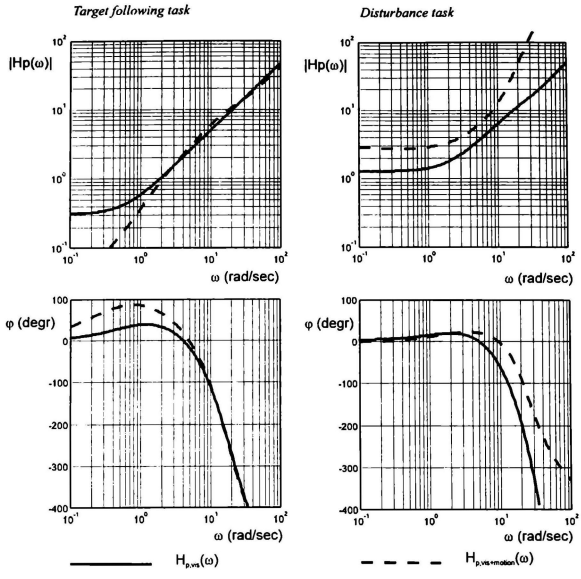


Figure 8. Bode plots of the pilot model transfer function for the target following task and the disturbance task with and without whole body motion.

$$H_{p,Foll}(\omega).H_c(\omega) = \frac{H_{vis}(\omega).H_I(\omega).e^{-j\omega\tau_{s,1}}}{\{1 + H_{MF}(\omega).H_{motion}(\omega).H_I(\omega)e^{-j\omega\tau_{s,2}}.H_c(\omega)\}}.H_c(\omega), \tag{3.3}$$

and for the disturbance task:

$$H_{p,Dist}(\omega).H_c(\omega) = \{H_{vis}(\omega).e^{-j\omega\tau_{s,1}} + H_{MF}(\omega).H_{motion}(\omega).e^{-j\omega\tau_{s,2}}\}.H_I(\omega).H_c(\omega). \tag{3.4}$$

The time delays for the PFD, the outside world display, and the motion system are assumed not to be equal, hence a difference is made between $\tau_{s,1}$ and $\tau_{s,2}$.

To adjust the pilot model, the sensory weightings are adjusted to obtain an adequate stability (phase margin) and an optimal tracking performance. The required change of the weighting factors and the increase of the tracking error, when changing from the control of the real aircraft to the control of the simulated aircraft, are a direct measure for the impact of simulation time delays and the washout filters on pilot's control behavior.

The transfer functions of the pilot in control of the simulated aircraft can be derived for the influence of the individual time delays τ_s of the PFD, the outside visual system, and of the motion system. Similarly, the influence due to the washout filters can be derived. In the analysis, the individual influence of the time delays and the washout filters on the closed loop performance and stability has been determined using the full pilot model of Fig. 4. The time delays used are τ_s = 50 and 150 ms. The classical washout filters for the simulation of the longitudinal attitude θ are applied, and the transfer functions of their low-pass and high-pass channels are [11]:

$$H_{high-pass}(\omega) = \frac{j\omega}{(j\omega + 1)}, \tag{3.5}$$

and

$$H_{low-pass}(\omega) = \frac{25}{(j\omega)^2 + 10j\omega + 25}. \tag{3.6}$$

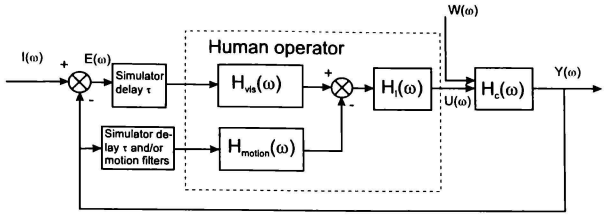


Figure 9. Simplified scheme of the pilot model, the simulation time delays and the washout filters in the control loop.

It was mentioned earlier that the simulator motion system which imparts the whole-body motion cues to the pilot has its own dynamic characteristics which introduces additional changes of the magnitude and phase of the simulated aircraft motions. Innovative approaches are becoming available to address this problem and, at least in research simulators, may be practically eliminated. For the purpose of this research, however, these will be left out of the analysis.

The aircraft dynamics, $H_c(\omega)$, used in the analysis are represented by a double integrator (K/s^2). These system dynamics were chosen for this first analysis because the stability of the control of this system is particularly sensitive for changes of the magnitude and phase of the open loop, the pilot-simulated aircraft combination.

Finally, the results of this analysis are presented next.

4. Results

The influence of the simulation time delays on the target following task and on the disturbance task are now shown, using the pilot perception model developed earlier. The dynamics of the classical washout filter are introduced to examine their influence as well on the performance of the pilot.

Following task.

As shown in Table 1, the influence of a time delay in the target following task is small for τ_s = 50 ms and moderate for τ_s = 150 ms. The delay of the PFD in the feed forward path of the control loop, eq. (3.5) appears to inject the most significant influence. On the other hand, delays in the feedback path of the control loop, in the peripheral visual cues and in the motion cues, have a minor influence on pilot's performance and adaptation. To adapt the model for the best possible performance, the weighting factor, $K_{C,a}$, of the attitude perceived from the PFD, had to be decreased. The standard deviation of the attitude error increased by about 40 % with feedback of peripheral visual motion cues and by 5 % with feedback of whole-body motion with the motion system. In Fig. 10, the response of the attitude error e_θ on a step input change of the reference attitude is shown. This figure clearly demonstrates the contribution of the time delay in the presentation on the PFD to the response of the closed loop.

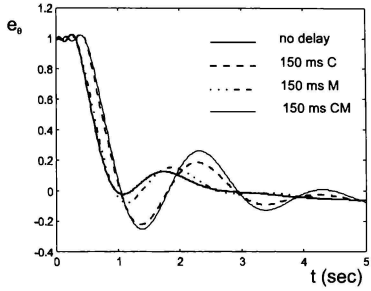


Figure 10. Step response of the attitude error e_θ to the reference attitude θ_{ref} due to time delays in the presentation of the PFD (C), the motion system (M) and both (CM).

Washout filters have a negligible effect on the performance and on the required adaptation of the pilot, since they only influence the feedback loop, eq. (3.5). This result corresponds with the fact that motion cues in the target following task influence pilot's control behavior in the low frequency range, Fig. 8.

Disturbance task

For the disturbance task, the influence of time delays is moderate for 50 ms, however it becomes quite significant for 150 ms, Table 2. This is primarily a result of the delay of the peripheral visual and motion

Table1. Summary of the results of the analysis of the influence of time delays and washout filters on pilot's control behavior and performance in the following task.

delay (ms)	$K_{C,a}$	$K_{C,r}$	$K_{P,r}$	K_{SCC}	Error sd (degr)	ω_c rad/sec	φ_m (degr)
PFD, Peripheral display, and no whole-body motion							
-	0.31	0.465	0.100	-	1.179	2.29	38.0
50 ms	0.31	0.465	0.100	-	1.254	2.37	30.8
150 ms	0.19	0.380	0.110	-	1.643	1.96	31.0
PFD, whole-body motion, and no peripheral displays							
-	0.31	0.465	-	0.300	1.094	1.83	70.2
50 ms	0.28	0.448	-	0.310	1.173	1.78	69.6
150 ms	0.25	0.40		0.275	1.271	1.84	56.2
PFD and motion with washout filters							
-	0.31	0.465	-	0.300	1.094	1.83	70.2
Washout	0.31	0.465		0.300	1.096	2.15	62.4

Table 2. Summary of the results of the analysis of the influence of time delays and washout filters on pilot's control behavior and performance in the disturbance task.

delay (ms)	$K_{C,a}$	$K_{C,r}$	$K_{P,r}$	K_{SCC}	Error sd (degr)	ω_c rad/sec	φ_m (degr)
PFD, peripheral display, and no whole-body motion							
-	1.46	0.20	0.38	-	1.38	3.21	17.4
50	1.20	0.17	0.54	-	1.58	3.38	16.4
150	0.62	0.112	0.465	-	2.96	2.61	16.3
PFD, whole-body motion, and no peripheral displays							
-	2.90	0.25	-	1.10	0.704	4.66	22.0
50	2.50	0.075	-	1.20	0.833	4.54	21.5
150	1.00	0.125	-	0.80	2.018	3.42	16.0
PFD, whole-body motion with washout filtering							
-	2.90	0.25	-	1.10	0.704	4.66	22.0
Washout	2.90	0.174	-	1.22	0.707	4.18	20.2

cues. In the disturbance task the peripheral visual and vestibular motion cues act upon the feed-forward path of the control loop, eq. (3.6). With the peripheral-visual feedback of motion and in spite of the adaptation of pilot's behavior, the error standard deviation increased by 115 % and for vestibular motion cues by 185 %. To compensate for the delay the pilot has to decrease his gain, resulting in a lower crossover frequency with approximately constant phase margin. The result of the time delay of the vestibular motion cues is approximately equal to deleting the motion cues altogether. This corresponds with the change in control behavior due to the addition of motion feedback, Fig. 8.

The influence of washout filtering is small to moderate. Due to the changes in the magnitude and the phase angle as a result of the high-pass and low-pass filters, the model had to be adapted by decreasing the crossover frequency at approximately constant phase margin.

5. Summary and conclusions

The model, described in this paper, accounts for the influence of central and peripheral visual and vestibular cues on the pilot's control behavior. The model was evaluated with the pilot's control behavior from a data base of measured transfer functions in the target following task and the disturbance task [7]. This model is able to describe the change in behavior when peripheral visual and vestibular motion cues are fed back to the pilot, both for maneuver motion and disturbance motion. The influence of peripheral visual and vestibular motion cues on pilot tracking behavior is essentially the same, although the effect of vestibular cues is stronger.

In this paper, the model described is used to investigate the influence of pure time delays caused by the simulation process, and due to the magnitude and phase distortion introduced by washout filters. The results indeed show that the pilot has to change

his control behavior in order to adapt to the simulated-aircraft characteristics. It was shown that the pilot's adaptation and the change in performance are small to moderate when the time delay is 50 ms, but moderate to large when the time delay is 150 ms. In the target following task the time delay of the presentation on the PFD has the strongest influence, however in the disturbance task, the time delays of the peripheral visual and vestibular motion cues has the largest impact. The pilot's adaptation and resulting changes in performance due to the classical washout filters are moderate and smaller than in the case of 150 ms pure time delay.

The results obtained from the analysis are primarily independent of the dynamic characteristics of the simulated aircraft, due to the way in which the motion cues contribute to the pilot's control behavior, as shown in eqs. (3.1) through (3.4).

These results demonstrate that a simulation time delay of 150 ms in the disturbance task forces the pilot to adapt his behavior considerably to maintain stability and obtain reasonable performance. It is questionable if in that case adequate transfer of training can be obtained, especially in the case of young inexperienced pilots. To avoid this a significant decrease of the simulation time delays is warranted.

The analysis shows that for a vehicle with double integrator dynamics a good impression of the influence of the shortcomings of simulation on pilot's behavior and performance can be found. Further analysis for real aircraft dynamics and actual simulator characteristics can provide information on possible improvements.

References

1. Advani, S.K., and R.J. Verbeek (1994). The influence of Platform Inertial Prperties on Simulator Motion System Performance. Proceedings of the Flight Simulation Technologies Conference, Scottsdale, August, 1994. AIAA-94-3418-CP.
2. Airplane Simulator Qualification. Advisory Circular AC No.:120-40C (draft). U.S. Department of Transportation, Federal Aviation Administration.
3. Hosman, R.J.A.W., and J.C. van der Vaart, (1978). Vestibular models and thresholds of motion perception. Results of tests in a flight simulator. Delft University of Technology, Department of Aerospace Engineering, Report LR-265.
4. Hosman, R.J.A.W., J.C. van der Vaart, and G.A.J. van de Moesdijk (1979). Optimization and evaluation of linear motion filters. Proceedings of the Fifteenth Annual Conference on Manual Control, AFFDL-TR-79-3134.
5. Hosman, R.J.A.W., and J.C. van der Vaart (1981). Effects of vestibular and visual motion perception on task performance. Acta Psychologica, 48 (1981) 271-287.
6. Hosman, R.J.A.W. and J.C. van der Vaart.(1988). Visual-Vestibular Interaction in Pilot's Perception of Aircraft or Simuator Motions. Fligt Simulation Technologies Conference, Atlanta, August, 1988. AIAA-88-4622-CP.
7. Hosman, R.J.A.W., 1996. Pilot's perception and control of aircraft motions. Ph.D. Thesis, Delft University of Technology. Delft University Press, ISBN: 90-407-1384-7. Delft, November 1996.
8. Levison, W.H., and A.M Junker (1978). A model for the Pilot's Use of Motion Cues in Steady-Sate Roll-Axis Tracking Tasks. AIAA Flight Simulation Technology Conference. Arlington, Texas
9. Martin Edward, A. 1995. Motion and Force Simulation Systems, I. Lecture notes from AIAA Flight Simulation Update-1995, State University of New York, Binghampton, N.Y..
10. McRuer, D.T. and E.S. Krendel (1974). Mathematical Models of Human Pilot Behavior. AGARDograph No. 188.
11. Nahon, M and L.D. Reid (1989). Simulator Motion-Drive Algorithms: A Designer's Perspective. JOURNAL OF GUIDANCE, AIAA, Vol. 13,No.2.
12. Paasen, M.M. van (1995). A Model of the Arm's Neuromuscular System for Manual Control. Preprints of the 6th IFAC/IFIP/IFORS/IEA Symposium on Analyses, Design and Evaluation of Man-Machine Systems. Massachusetts Institute of Technology, Cambridge, Ma, June 27-29,1995.
13. Rasmussen, Jens, 1983. Skills, Rules, and Knowledge; Signals, Signs, and Symbols, and Other Distinctions in Human Performance Models. IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS, Vol. SMC-13, No. 3, May/June 1983.
14. Ray, Paul A., 1996. Quality Flight Simulation Cueing. Why? AIAA Modeling and Flight Simulation Technology Conference, 1996. AIAA Paper 96-3488-CP.
15. Shirley, R.S., and L.R. Young (1968). Motion cues in Man-Vehicle control. Effects of roll-motion on human operator's behavior in compensatory systems with disturbance inputs. IEEE Transactions on Man-Machine Systems, Vol MMS9, No. 4.
16. Wickens, C.D. and J. Flach (1988). Information Processing. In: E.L. Wiener and D.C. Nagel (Eds.). Human Factors in Aviation. Academic Press, N.Y.

A NEW APPROACH TO SENSORY INTEGRATION MODELING

Gary R. George
Graduate Student

Frank M. Cardullo
Associate Professor of Mechanical Engineering

Department of Mechanical Engineering
State University of New York
Binghamton, N. Y. USA

Abstract

This paper describes the general methodology and available advanced techniques that are required to develop a unified model which integrates the responses of transfer functions representing each of the separate sensory modalities. These methods include Kalman filtering, federated Kalman filters, neural networks and various fuzzy techniques to implement the integration process. Much individual sensor modeling has been developed by other researchers over the years and is based on psychophysical data from that time frame. Some work has been done using a classical Kalman filter to model the sensory integration process for the sensory modalities. A newer approach is federated Kalman filters which has been proposed to deal with a similar problem in avionics where an aircraft flight control computer may receive data from several different types of sensor systems. This approach uses a bank of Kalman filters in parallel.

The more advanced approach being pursued by the authors considers the use of fuzzy logic and fuzzy measures to model the sensory integration process. Fuzzy sets do not possess sharp distinctions, capture vagueness of concepts in natural lanquage and capture measurement uncertainties as found in human sensory systems. Fuzzy measures on the other hand is an outgrowth of classical measure theory and must not be confused with fuzzy logic or fuzzy inferences. Fuzzy measures capture lack of information resulting from nonspecificity of evidence and/or conflicting evidence as would be associated with the human sensory system. Normally the definitions of the fuzzy rules and possibility measures are done by a subject matter expert. This paper will also discuss the use of neural nets to cluster the data and the results employed to define the necessary fuzzy parameters for the fuzzy system.

The objective of this research is to develop a tool which can be used to define the cuing requirements and metrics for systems employing virtual environments such as flight simulators.

Past Research

There have been a number of attempts to employ physiological modeling to define simulator motion cueing requirements. The first attempt was by Young et al (1976). There have been other researchers who have followed this approach, namely; Forrstrom, Doty and Cardullo (1986), Bussolari et al (1989), Hosman et al (1979), Borah (1978), and Zacharias (1978) etc. Professor Lloyd Reid of the University of Toronto developed a software package for the National Advanced Driving Simulator (NADS) Motion Base Sizing Study. This software is grounded in the principle of minimizing the perceptual error between the simulator motion system and the actual vehicle. The integration of several sensory modalities into a common perception of motion is an important part of realistic physiological modeling. Previous attempts to integrate the sensations from the various motion sensory apparatus have centered about conventional Kalman filter approaches Borah (1978), Zacharias (1978)). This technique has some shortcomings such as; a priori knowledge of the various levels of subcognitive and cognitive processing in the brain is required to develop the state equations and the technique does not account for learning that occurs when humans perform tasks. An area in all the previous work which has generally been neglected is the effect on posture of the human body due to the vehicle and the motion system dynamics and the consequent effect on motion perception.

Consequently, some other techniques should be investigated such as neural networks, fuzzy techniques and federated filters to expand the modeling process particularly the integration phase. The federated filter technique developed by Carlson (1990) is being employed in avionics systems to resolve ambiguities in data from several sensor systems. Some work has been done by experimental psychologist Massaro (1990) in using fuzzy inferences to model the integration of auditory and visual stimuli. Other than this work by

Copyright © 1997 by the American Institute of Aeronautics and Astronautics, Inc. All rights reserved.

Massaro and the previously mentioned work in Kalman filtering little modeling of the general sensory integration process of sensory modalities appears in the literature using modern integration techniques.

The modeling of the various sensory modalities has been presented by Gum (1973) as s domain transfer functions with nonlinearities such as thresholds and dead bands. Gum describes step and frequency responses to these models. For example as presented in the Gum paper the semicircular canals can be considered to be a highly overdamped torsional pendulum with central nervous system latency, adaptation and a threshold as illustrated in Figure 1 The input is angular rate in degrees/second squared with the output of afferent firing rate (AFR) in impulses per second.

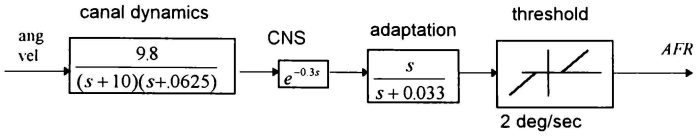


Figure 1 Semicircular Canal Model

Another important consideration is the orientation of the human body resulting from external forces. A total model of the human body is necessary. A biomechanics model will provide the dynamics of the entire body and the reaction of it to external forces. The equations of motion are derived using classical energy methods. The primary purpose of this biomechanics model is to provide limb and torso dynamics as inputs to the muscle and tendon models. The orientation of the head relative to the body and the environment is important for visual and vestibular responses. The head neck model should be incorporated in the total biomechanics model. Current approaches to this type of human perception modeling do not involve these dynamics. However, it is felt by several researchers that the conscious and subconscious control of posture is a major factor in vehicle control strategies. The effort to control forces in a turn to maintain posture is experienced by anyone who drives an automobile for example. Hence, it is felt that this element is a necessity to be included.

Integration of Sensory Inputs

Inputs from the sensory systems must be integrated in the nervous system to develop a perception of orientation and motion. This process can be modeled in several ways. The various fuzzy integration approaches that are applicable are described below. Fuzzy set theory, fuzzy measure theory (evidence and possibility theory) as well as classical set and probability theory (e. g. Kalman filtering) form the mathematical basis of uncertainty theory. There are relations between many of these. For example classical set theory is subsumed under fuzzy sets while probability theory is a part of evidence theory. Evidence theory is really a part of fuzzy measure theory. Detailed mathematical details of these subjects can be found in texts by Wang et al (1992), Kosko (1993) and Klir & Folger (1988). Klir (1993) presents a very detailed overview of the latest developments in all these subjects. Fuzzy systems which employ mappings between fuzzy sets are becoming increasingly popular particularly in the Eastern World with several applications in household products, control systems, data systems, decision making, pattern recognition, reliability theory, medical diagnostics, experimental psychology and providing rules for neural networks among others. Some work in fuzzy measures using Schafer- Dempster evidence theory has been applied to target recognition using multiple sensor inputs in avionics systems with success. Previously Massaro (1990) was mentioned for his work in the fuzzy theory of perception in the world of psychology which will be discussed later.

A distinct advantage of fuzzy systems is that they are model free estimators. They estimate a function without requiring detailed mathematical functions of how the output depends on the input. The fuzzy parameters or rules can be learned from examples or experience of a subject matter expert or the use of neural nets examining data to develop trends or clusters in the data. Modeling using this approach based on the data results in a system that has the necessary inherent nonlinearities and complexities that would be impossible to incorporate in other approaches such as a Kalman filter. For these reasons fuzzy techniques appear to be a highly desirable technique to

model the extremely complicated human sensory integration to perception problem described previously.

Fuzzy Set Theory

With classical set theory the concept of a crisp set is used where there is a distinct and precise boundary. Fuzzy sets have boundaries that are not precise in that the transition from nonmembership to membership is gradual rather than abrupt. This gradual change is defined by a membership grade function μ_A

$$\mu_A : X \longrightarrow [0,1].$$

where X is the universal crisp set that is being considered and A is the label of fuzzy set defined. The value of the membership grade is between 0 and 1 thus defining the compatibility to the set in question. The basic features of fuzzy sets is the lack of sharp distinctions, the ability to capture the vagueness of natural lanquage and it captures measurement uncertainty. For example consider Figure 2 from Klir (1993) which defines five fuzzy sets (A_1 .....A_5) on a universal closed interval 1 to 100. It should be noted that the shapes of the fuzzy functions can be arbitrary and can assume multidimensions on X_i forming hypershapes. The fuzzy sets shown represent linguistic terms and ideas associated with very small, small, medium, large and very large. This is an important advantage since the system description can be described in a natural manner which is important for investigators that do not have mathematical backgrounds. For example in Figure 2 the membership grade of V_1 in the fuzzy set very small is approximated by membership grade .6 and .33 for fuzzy set small, while all others are zero. This a good example of being able to generalize concepts of intervals of real numbers and real numbers to fuzzy intervals and fuzzy numbers. The advantage of the concepts of fuzzy numbers and fuzzy intervals is the ability to express the effect of measurement error more faithfully in a simple linguistic manner which in their own way are vague. The illustration in the left of Figure 3 is another representation of fuzzy sets.

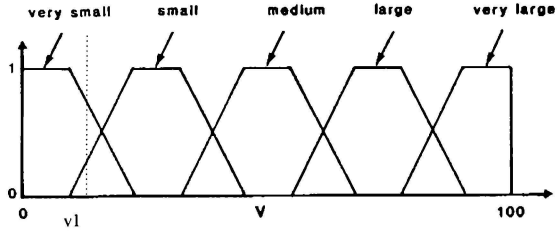


Figure 2 Fuzzy set

When fuzzy numbers and intervals as defined above represent the state of a system that system is known as a fuzzy system. The necessary steps in defining a fuzzy system include selection of input and output variables, selection of the linguistic states for each variable, selection of linguistic fuzzy inference rules, initial definition of membership grade functions for all linguistic states, initial selection of operations for inference rules, selection of a defuzzification methodology and final tuning either manually or with neural networks. There are a number of fuzzy operators such as multiplication, addition Klir & Folger (1988) that allows fuzzy relationships to be developed to describe the system. The fuzzy inference rules are called fuzzy associative memories (FAM's) by Kosko (1992). The key part in developing a fuzzy system is to define the fuzzy functions (shape and linguistic label) based on practical experience or the examination of test data. Normally this is done by a subject matter expert or initially by best guess and then tuned based on system performance. Kosko (1992) has suggested that test data can be input to neural networks and the fuzzy rules developed from clustering of patterns in the data. This concept known as fuzzy neural networks eliminates the need for a specific subject matter expert and thereby accounts for learning.

Figure 4 is a block diagram of a fuzzy system showing the FAM architecture. Much work has been successfully applied to control systems with this approach. Basically the fuzzy system maps fuzzy sets to other fuzzy sets. In this case the fuzzy set x is mapped to fuzzy set y via several FAM rules. A typical Fam linguistic rule might be " IF X IS MODERATELY LARGE THEN Y IS VERY SMALL". The membership grades associated with these rules are then summed over their intervals to develop a final fuzzy set. In many cases the final output is from a centroiding approach to define a final measure.

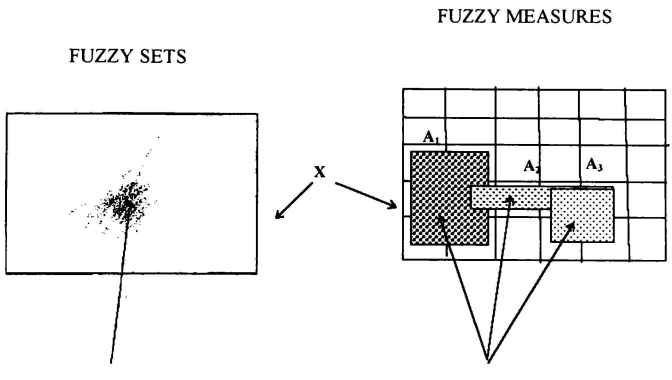


Figure 3 Fuzzy sets and fuzzy measures

Figure 5 shows a very simple perceptual fuzzy system where angular acceleration fuzzy set is mapped to fuzzy set afferent responses. In a more rigorous perceptual model there would several of these mappings of which many could be multidimensional. The linguistic labels have been defined for each triangular fuzzy set. The shapes of the fuzzy sets is at the discretion of the designer. The FAM rules then provide the mappings between the sets. These rules or fuzzy patches provide an overlap of the data to approximate the system. It is apparent that sloppy rules result in big patches. These patches cover a nonlinear relationship between acceleration and afferent responses. As the patches become smaller i.e. the triangular membership function become spikes the exact nonlinear relationship is revealed. The penalty here is the large number of rules required. The fuzzy system then guesses at equations that are too nonlinear or complex to analytically derive. The size of the fuzzy patches or rules provides a means for that approximation to be done at different levels of abstraction.

Consider a simple example of how this process would work for an angular acceleration a1 which is in the overlap of FAM rule 1 and 2. FAM rule 1 is "IF ACCELERATION IS LOW THEN MOTION PERCEPTION IS LOW" and FAM rule 2 is "IF ACCELERATION IS MEDIUM THEN MOTION PERCEPTION IS MEDIUM". Both rules fire but to different degrees as defined by the membership grade. Rule 1 fires to extent .6 (m1) while rule 2 fires to extent .15 (m2). The end result is an afferent response that is a new fuzzy set made up of .6 of the afferent fuzzy set perception low and .15 of afferent fuzzy set perception median. By taking the centroid of the

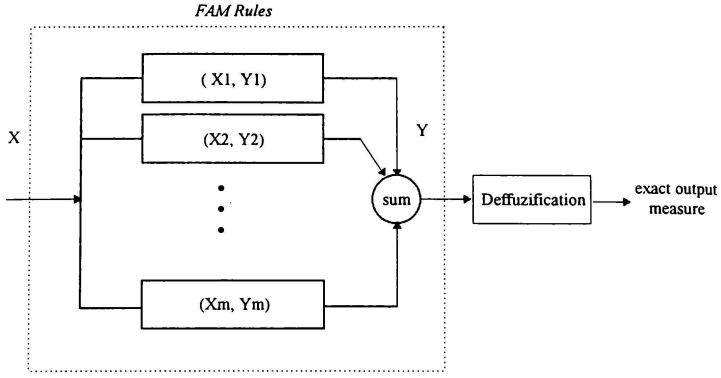


Figure 4 Block diagram of fuzzy system

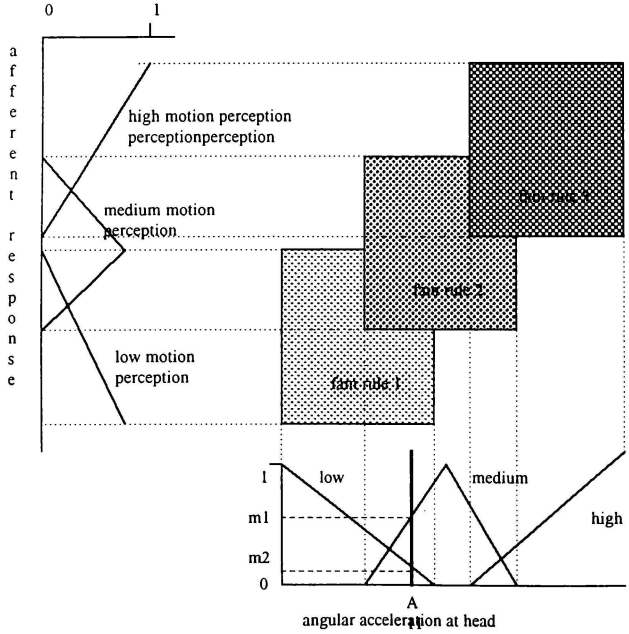


Figure 5 Fuzzy set mappings

resulting two overlapping sets here the defuzzification process gives an estimate of afferent firing rate.

Application to the unified sensor model would include the extensive development of multidimensional fuzzy rules and relationships. The example above represents a very simple application for illustration of the process. Starting points for these rules might be linearized models of various sensory modalities and the how these models differ from actual test data. Depending on the available psychophysical data, neural networks might be used to develop the fuzzy rules.

Fuzzy Logical Model of Perception (FLMP)

Another part of fuzzy thinking includes fuzzy truth values where the propositions are not distinctly true or false as found in Boolean logic. Massaro (1992) developed the FLMP based on the use fuzzy-truth values. These values lie between 0 and 1 corresponding to a proposition being completely true or false. The value of .5 is a completely ambiguous situation. Continuously valued stimuli are evaluated, and matched against prototype depictions in the computer memory. The identification decision is made then based on the relative goodness of match of the stimuli information with the relevant prototype descriptions or propositions. This matching is done using the fuzzy-truth values. Feature integration consists of multiplying the feature or stimuli values supporting a particular alternative A_k. Thus if x_k and y_k are the values supporting A_k then the total support a_k for the alternative A_k would be simply the product of x_k and y_k .

It is desirable then to develop the relative degree of support for each alternative. As defined by Massaro this is the probability of a response A_k given the specific stimuli X_i and Y_i .

$$P(A_k \mid x=x_i \ , y=y_i) = \frac{x_k y_k}{\sum_{i=1}^{m} x_i y_i}$$

The denominator represents the sum of merit of all m relevant alternatives. The evaluation of the perceptual measure for each source of information is represented by a truth value indicating the merit of a particular alternative. The integration is then the multiplicative combination of the associated truth values.

Massaro's original work was finely tuned against test data and was found to be a good integrator of stimuli to predict responses of words from visual and auditory sources. In order to use this concept for the motion perception problem a large number of propositions of a prototype model would be required and various truth values assigned. This is the first work in psychology that models sensory inputs to a perception measure not using classical signal detection theory such as found in Macmillan (1991).

Fuzzy Measures

Fuzzy measures is an outgrowth of classical measure theory and must not be confused with fuzzy logic or fuzzy inferences. The two illustrations in Figure 3 illustrates the difference between fuzzy logic and fuzzy measures. In the fuzzy measures illustrated on the right of Figure 4 the sets are considered to be crisp and we would like to define the likelihood of membership in each of these sets. The $g_x(A_i)$ is the degree of evidence (belief, plausibility, probability, ect.) that x, which incompletely characterized, belongs to A_i. Fuzzy measures capture lack of information resulting from nonspecificity of evidence and or conflicting evidence.

An area of fuzzy measures that is applicable to the unified sensory model is belief functions. There has been an interest in developing multiple sensor integration for avionics and target detection systems using fuzzy measures. In reviewing the literature there have been applications of fuzzy logic and measures to multiple sensor integration. Early work by Bogler (1987) used Shaffer-Dempster approaches with belief functions. The technique requires that basic probability assignments be assigned to various sets and subsets of stimuli input. Shafer-Dempster reasoning (evidence theory) is also a generalization of Bayes reasoning that allow confidences (known as basic probability assignments) to be assigned to sets of propositions rather than to mutually exclusive propositions. The information from each sensor may not be presented at the same level of abstraction e.g. the vestibular input represents acceleration as the visual might represent certain patterns. Shafer-Dempster reasoning can accurately and efficiently manipulate information at various levels of abstraction. It is particularly powerful combining evidence from separate sources. From Klir (1989) the standard Shafer-Dempster rule of combinations of evidence from independent sources is:

$$m_{12}(A) = \frac{\sum_{B \cap C = A} m_1(B) * m_2(C)}{1 - K}$$

where

$$K = \sum_{B_i \cap C = 0} m_1(B) * m_2(C)$$

for

$A \neq 0$ and $m_{12}(0) = 0$.

The m values are the basic probabilities that are assigned the sets. These must be done by someone who is familiar with the problem that is being modeled.

Consider a simple example of sensory integration using the Dempster's rule for combining evidence from separate sensory sources. Denote p_1 to be the vestibular angular rate above threshold for motion basic probability assignment and p_2 to be one of visual input basic probability assignments for the perception of yaw motion. We consider only two stimuli out of all the possible ones for simplicity. These values would be determined based on the type of motion cueing and visual display and defined by a subject matter expert. Let us say that the simulated visual imagery was good in content, contrast and general quality while we have a large motion base to provide good vestibular cues. It therefore seems logical that the p_1 might be .8 since yaw visual cueing would be adequate and the p_2 might also be .8. Further assume that in this case visual input is a subset of vestibular. Mathematically this can be expressed as:

$m_1(visual)=0.8$ $\quad$ $m_1(all)=0.2$
$m_2(vestibular)=0.8$ $\quad$ $m_2(all)=0.2$
$visual \subset vestibular \subset all$

We therefore have a very simple nest of evidence. Using Dempster's rule of combining p_1 and p_2 the resulting product is:

$$m_1 * m_2 = \begin{vmatrix} m(vestibular \cap visual)=.64 \\ m(vestibular \cap all)=.16 \\ m(all \cap visual)=.16 \\ m(all \cap all)=.04 \end{vmatrix}$$

Now note due to the subsets we have defined the following that can be defined:

vestibular $\cap$ visual = visual
vestibular $\cap$ all = vestibular
all $\cap$ visual = visual
all $\cap$ all = all

where $\cap$ denotes set intersection. So the final values are as follows:

$$m_1 * m_2 = \begin{vmatrix} m(visual)=.80 \\ m(vestibular)=.16 \\ m(all)=.04 \end{vmatrix}$$

To conclude then about this combination of evidence we see that the support of the motion being perceived via the visual is .8 but the support for it being perceived by the vestibular is .96 (the sum .16+.8).

The following was a very simple example of combining nested information. In the more involved perceptual modeling there would be many overlapping as well as disjoint evidence to be considered and combined by the full Dempster's rule of combination. A key factor here will be the assignments of the basic probabilities m_i and the nesting architecture of the sets as it relates to psychophysical data.

Example

A simple example will illustrate the fuzzy set concept of modeling sensory integration in the motion perception process. Figure xxx is a fuzzy associative memory (FAM) that maps afferent firing responses of vestibular and somatosensory receptors to the classification of the motion perception metric. The source of this data could be from transfer functions that model the sensory functions as we have previously illustrated in Figure 1. Inputs to these nonlinear transfer functions would be accelerations or specific forces that are applied to the human body. The afferent firing rates for each are broken into three fuzzy sets with labels PS positive small, PM positive medium and PL positive large over the two input universes of discourse. The modeling of sensory integration is in the resulting FAM mapping of SM small perception, MP medium perception and LP large perception as indicated in the squares. This mapping definition can be done by subject matter experts or the examination of test data using unsupervised neural networks to uncover clustering in the various cells. In particular when we consider further inputs into the sensory integration process and the size of the FAM table increases the neural net approach becomes desirable. We will assume that the shape of the membership grades as symmetric triangles centered at their respective values although they could be other shapes depending on how the data clusters in each FAM cell. Another issue is the amount of overlap among the contiguous fuzzy sets represented by the triangular membership grades. Too much overlap blurs the distinction between fuzzy set values. To little overlap produces instability. A good rule of thumb is to start with 25% overlap.

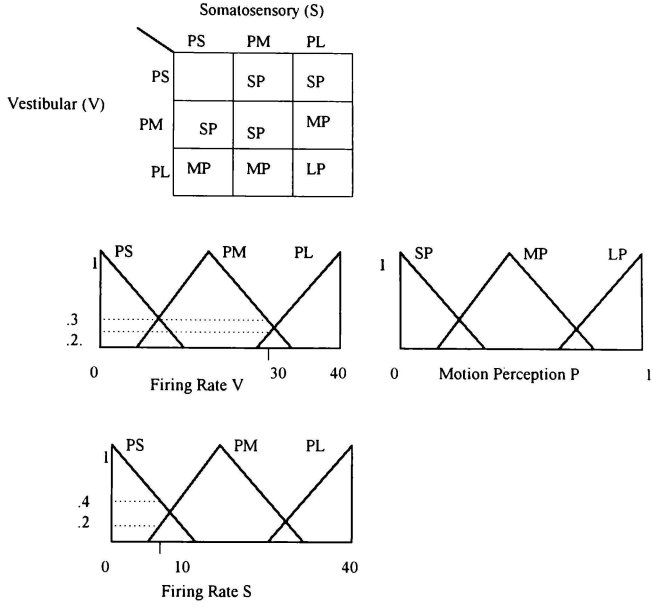


Figure 6. Fuzzy Associative Memory and Membership Grades

The FAM in Figure 6 represents eight rules as indicated by the labeled eight cells. As can be seen some cells are completely empty indicating that no rules apply here. The set of all possible input-output (V, S, F(V, S)) represent the sensory integration product space, in this case in R^3. Let us assume that the afferent firing ranges from 0 to 40 spikes per second for both vestibular and somatosensory and the motion perception metric ranges from 0 to 1 where 1 indicates high motion perception and 0 none. The FAM table then supplies a representation of rules as they apply to the sensory integration. Lets consider a afferent vestibular input of 30 and a somatosensory input of 10. Examination of the membership grades determine that the vestibular response is more PM (grade of .3) than it is PL (membership of.2). In a similar manner it can be seen from figure 6 that the

somatosensory response is more PS (grade of .4) than PM (grade of .2). These particular values chosen:correspond to two rules as seen in the FAM chart:

IF Vestibular Reponse is PM And Somatosensory Response is PS then Motion Perception is SP
IF Vestibular Reponse is PL And Somatosensory Response is PM then Motion Perception is MP

The sensory integration follows from fuzzy logic operators of the conjunctive AND which combines the antecedent fit values with the minimum. This combined fit value then scales the corresponding fuzzy set with pairwise minimums, that are applied to the perceptual fuzzy intervals. The system then adds the minimum-scaled output fuzzy sets and then computes the fuzzy centroid of the output of perception. Figure 7 illustrates the procedure for our simple example. The pairwise minimum of the top vestibular and somatosensory input is .3 while the lower input combination is .2. These two conjunctive AND values can now be applied to the perceptual metric fuzzy sets of SP and MP which now form a mapped output of perception as seen in figure 8. To determine the value of the perceptual motion metric a defuzzification is necessary which is accomplished by finding the area centroid of the two areas which in this case is a perceptual motion metric of approximately .325.

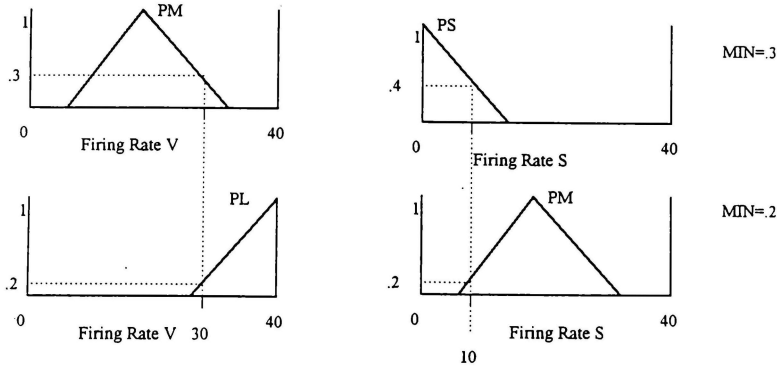


Figure 7. FAM correlation-minimum inference procedure

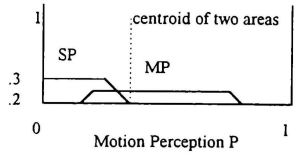


Figure 8. Scaled Perceptual Motion Output for SP and MP

Future Research

The direction of the research is to develop a necessary data base of information on the separate sensory modalities and the integration into a common perception. This will include psychophysical data and the use of simplified models. With this the FAM's and membership grades can be assigned to fuzzy systems to develop a unified theory of motion perception. Further work will be done in assigning the basic probabilities to apply sensor fusion or integration using evidence theory.

Conclusions

This paper has discussed the necessity of a unified sensory model for simulation and virtual reality cuing device development. The problem represents a highly nonlinear and complicated system that is not easily modeled using standard techniques. Other than some earlier work using Kalman filters little has been done to develop a unified sensory model. Work in the use of fuzzy propositions has been applied to some psychology work mentioned in the paper.

Since the process is difficult to model the use of modelless estimators seems reasonable. These systems include neural nets and fuzzy techniques. In particular the use of fuzzy systems and the mapping between fuzzy sets offers a unified theory of perception that includes the nonlinear and complicated aspects of the human integration process. The rules can be based on neural network clustering of test data or simplified linear models of sensory modalities. The use of linguistic labels is highly desirable for engineers interfacing with subject matter experts that may not have strong mathematical backgrounds. Although the use of fuzzy systems seems an ideal approach the use of fuzzy measures which is a considerably different shows promise. In particular the use of evidence theory Schafer-Dempster recombination rules show promise in sensor fusion and should be considered.

This work has other applications than simulation and virtual reality such as telerobotics and a wide range of experimental psychology. A generalized technique may result that could applied to experimental data of different varieties and conclusions drawn on human perception and behavior.

References

Borah, J. , Young, L. R., Curry, R. E., "Multisensory perception model for application to aircraft simulation." 11th NTEC/Industry Conference, 1978.

Bussolari, S. et al. "The use of vestibular models for design and evaluation of flight simulator motion." AIAA Flight Simulation Technologies Conference, 1989.

Carlson, N. A., 'Federated Square Root Filters for Decentralized Parallel Processes", IEEE Truncations on Aerospace and Electronic Systems, Vol. 26, No.3 May 1990.

Forrstrom, K., Doty, J. and Cardullo, F. "Using human motion perception models to optimize flight simulator motion algorithms." AIAA Simulation Technologies Conference, 1985.

Gum, D. "Modeling of the human force and motion sensing mechanisms." AFHRL-TR72-54, 1973.

Hossman, R., Van der Vaart, J., van de Moesdlijk, G. "Optimilization and evaluation of linear motion filters." Fifteenth Annual Conference on Manual Control, March 20-22, 1979. Wright State University, Dayton, OH.

Klir, G. J. and Folger, T. A., Fuzzy Sets, Uncertainty, and Information, Prentice-Hall, 1988.

Klir, G. J.," Developments in Uncertainty-Based Information", Advances in Computers, Vol. 36, 1993.

Kosko, Bart, Neural Networks and Fuzzy System, Prentice-Hall, 1992.

Kosko, Bart, Fuzzy Thinking, Hyperion, N. Y., N. Y., 1992.

Massaro, D. W., Friedman, D., "Models of Integration Given Multiple Sources of Information", Psychological Review, Vol 97, 1990, Pg. 225-252.

Wang, Z, Klir, G. J. , Fuzzy Measure Theory, Plenum Press, N. Y. ,N. Y., 1992.

Young, L. R. et al. "A motion sensing model of human perception for simulator planning." 9th NTEC/Industry Conference, November 9-11, 1976.

Zacharias, G. "Motion cue models for pilot-vehicle analysis." AMRL-TR-78-2, May 1978.

THE HUMAN FACTORS OF FMS USAGE IN THE TERMINAL AREA

Katharine K. Lee*†
NASA Ames Research Center
Moffett Field, CA

Beverly D. Sanford†
Sterling Software, Inc.
Redwood City, CA

Rhonda A. Slattery‡
NASA Ames Research Center
Moffett Field, CA

Abstract

The development of advanced automation for arrival aircraft into the terminal area is being investigated for both the air traffic control (ATC) and airborne environments. For the automation to be effective and provide the best advisory information, aircraft trajectories must be accurately estimated. One way to enhance the aircraft's adherence to the trajectories assumed by advanced ATC automation is to supply trajectory information to the pilots. Such information is currently provided by ATC to pilots through voice communication. Providing the advisories to the pilots via their Flight Management Systems (FMS) could help improve the timeliness of the advisories, especially in a time-critical terminal area environment. To successfully implement the FMS for use in the terminal area, human factors issues such as pilot workload and the usability and understandability of the information presented via the FMS must be addressed. This study examines the human factors aspects of FMS usage strategies by commercial airline crews flying a Boeing 747-400 full-motion simulator in a terminal area flight segment. Results indicate that FMS-guided flight in the terminal area creates significantly higher amounts of individual crewmembers' head-down time, and increased levels of self-reported workload compared to conventional navigational means.

Introduction

Current development of automation in the air traffic control (ATC) terminal area includes the Center-TRACON Automation System (CTAS).[1] Such automation requires the accurate estimation of trajectories in order to help provide useful and efficient traffic advisories and accurately predict arrival times of the aircraft in the system.[2]

One way to enhance the aircraft's adherence to the trajectories assumed by the advanced automation is to supply trajectory information to the pilots. Trajectory information is traditionally provided to the pilots by voice advisories from ATC. This would be the conventional means to ensure compliance with the trajectories and help to realize the benefits of increased fuel efficiency and reduced delays that are part of the CTAS automation. Another way to provide these trajectories to the aircraft is directly to the aircraft's flight management system (FMS). The time-critical nature of the terminal area may be an environment in which to enhance pilot accuracy by utilizing the FMS. The FMS would provide more complete routing information in advance of when the pilots would normally receive such information. There is the possibility, however, that providing such information would be too confusing or overwhelming for the pilots to incorporate into their flight planning and control activities.

The experiment described in this paper was conducted with three objectives: 1) to examine the variability of aircraft maneuvers for aircraft equipped with on-board automation; 2) to determine the impact of pre-stored FMS routes on cockpit crews flying in the terminal area (thus providing validation for the trajectories produced by the CTAS automation); and 3) to determine acceptability of utilizing the FMS in the terminal area. The results from the first objective are described in Reference 3. The second two objectives are examined primarily from a human factors viewpoint, and are the focus of this paper. "Impact" is defined by measuring the effects of using the FMS on pilot workload, describe how the FMS was utilized, and how the FMS was incorporated into flight planning activities and other cockpit operations. These questions are especially

---

*This work was conducted as an employee of Sterling Software, Inc., Redwood City, CA.

† Human Factors Engineer.

‡ Research Scientist. Member AIAA.

Copyright © 1997 by the American Institute of Aeronautics and Astronautics, Inc. No copyright is asserted in the United States under Title 17, U.S. Code. The U.S. Government has a royalty-free license to exercise the copyright claimed herein for Governmental Purposes. All other rights are reserved by the copyright owner.

interesting given the fact that the FMS was not initially designed for use in the terminal area.

The benefits of the FMS can be obtained if two assumptions are met. First, the FMS database must be accurate. Second, from a human factors standpoint, it is critical that the FMS technology is easy to use under the constraints of terminal approach operations and is also acceptable to the pilots. Information should be easy to access, and inputs and changes should be achievable with minimum complexity. The means of conveying information from ATC to the cockpit crew, and the transfer of this information from crew to the FMS should be clear and concise. Within the cockpit, both the pilot flying (PF) and the pilot not-flying (PNF) should be aware of the planned flight route that is in the FMS, and what steps are needed to make changes and to monitor changes to that flight route.

When making changes to the flight systems in a conventionally-operated cockpit (not using the FMS), a pilot's actions are reflected in the physical setting of dials and switches, as well as the motions of reaching and turning. One pilot can be made aware of the other crewmember's actions by associating the movements with specific locations of equipment within the cockpit. If the cockpit tasks are distributed such that the PNF makes the majority of the operational changes to the aircraft systems as well as ATC communications,[4] the PF is still able to understand and observe the actions and settings of the PNF without taking attention away from the flying task. In contrast, using the FMS creates a situation in which changes can be made without obvious external feedback. Unless there is explicit discussion between the pilots during input of navigation settings in the FMS, there are no observable indications that modifications have been made to the pilot who did not implement the changes.

In such an operating environment, should the PF wish to verify or access the flight route changes implemented by the PNF, attention would need to be diverted from flying to interact with the FMS to access the desired information. Another layer of difficulty is therefore added to the flying task. In the terminal area environment, where there is little time to input and evaluate clearances, the crew has less opportunity to assess changes to the flight plan and initiate and evaluate those changes. Thus, in order to use the FMS in the terminal area, the PF needs to be satisfied with the possibility of not having adequate time to confirm modifications made to the flight plan within the FMS. If this is the case, it might be expected that more time is devoted to discussing the overall flight plan between the crewmembers. FMS usage should also be reflected in more "head-down" time, or time spent referencing, programming, or reviewing information on the Control Display Unit (CDU), which is how the pilots interface with the FMS.

A previous study[5] reported that flight crews using the FMS saw the work of their flying tasks to be greatly eased, but the FMS was more difficult to use during low altitude phases of flight, such as in the terminal area. Previous research[4] has also found that in simulation, the pilot flying sometimes took the initiative to make inputs into the FMS.

In this study, commercial airline crews flew a Boeing 747-400 full-motion simulator in a terminal area segment of flight under conditions of manual flight, autopilot, and autopilot coupled with FMS. These three conditions represent differing levels of automation available to aircraft today. The influence of the FMS on the cockpit crew's flight planning activities and the effects upon pilot workload were examined. Different FMS-use strategies were documented.

Within this experiment the crews were provided with specific scenarios and their response to FMS route changes in the terminal area were documented. The data collected in this study includes head-down time, FMS programming strategies, flight planning discussion, self-reported workload responses, and a description of the distribution of responsibilities within the cockpit. It is hypothesized that a great deal of time spent in a head-down mode would be associated with less time devoted to routine flight tasks. The FMS programming methods that are observed may determine how the FMS is used by flight crews, and indicate pilot expectations with regard to how they would prefer to use the FMS in the terminal area. The discussions about the flight plan also could indicate how much information the crew had about their flight route. Extensive discussions regarding the flight plan could indicate a greater understanding of the flight plan, as well as problems encountered in the flight planning process. Self-reported workload should provide a more direct indication of how the crews perceive they are impacted by FMS usage in the terminal area. Finally, increased head-down time should also affect how the crew duties are distributed in the cockpit.

Methods

The simulated flights took place in a full-motion, Boeing 747-400 simulator at NASA Ames Research Center's Crew-Vehicle Systems Research Facility (CVSRF). The simulator includes a high-fidelity visual system and simulated ATC communications (see Figure 1). A full description of the simulator and its avionics capabilities and specifications can be found in References 6 and 7. A total of ten, two-person commercial airline crews, each crew consisting of a Captain (CA) and a First Officer (FO), flew simulated terminal area approach segments. The two crew

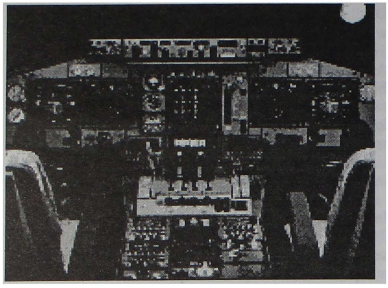


Figure 1. CVSRF 747-400 Cockpit

members alternated the flying responsibilities. Twelve simulated runs were scheduled for each day; due to equipment problems, one crew flew only 10 runs, for a total of 118 runs over all of the crews. The crews provided demographics data regarding their amount of flight experience and familiarity with the FMS and advanced automation as a whole.

After a training run, in which the flight plans and routes and basic FMS functionality were reviewed, the crews flew the terminal area arrival segment into the Dallas/Fort Worth (DFW) Airport. The flights' origination points alternated between the two East side meter fixes, Blue Ridge (see Figure 2) and Scurry (see Figure 3).

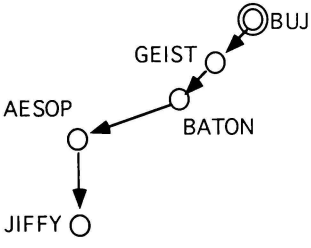


Figure 2. Blue Ridge FMS Arrival, BATNE1.

The CVSRF is capable of providing simulated background communications of other aircraft in the terminal area; these, in addition to ATC-issued clearances to the flight crew, were provided during each simulated flight. The crews flew the terminal arrival trajectories under three different flight conditions: manual flight (hand-flying using the flight director), autopilot (the typical way in which jet aircraft are currently flown into the DFW terminal area), and FMS

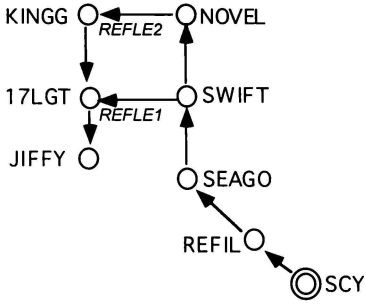


Figure 3. Scurry FMS Arrivals, REFLE1 and REFLE2.

(in which the autopilot was coupled with the FMS). When the simulated flight originated from the southeastern cornerpost fix, where the flight route was slightly longer than from the northeastern cornerpost fix, the crews were given a route amendment (REFLE2) to their original route clearance (REFLE1).

The simulated flights were videotaped, from a viewpoint behind the cockpit crew. After each simulated flight, the crews were debriefed and asked to provide ratings of their workload and contributors to their workload. Engineering data were collected on duration of flight, altitude, speed, and turn initiation; a report on pilot variability under the three flight conditions is described in Reference 3.

Each of the simulation videotapes was transcribed from the initiation of the flight to the touchdown on the runway. The transcripts included conversations between the cockpit crew and between the crew and ATC. Only conversations of an operational nature were transcribed. Due to recording problems, there were 6 runs that were not available for analysis.

The transcripts were coded to determine key events, noting the times within the simulation when certain events occurred, and where appropriate, the duration. The event data was then summarized and compared across the three flight conditions. Among the data collected from the transcripts/videotapes were: how the crews utilized/programmed the FMS (as determined by their communications), head-down time involved in using the FMS, and discussions about the flight plan.

The resulting data consisted of the following: questionnaire data and crew comments, discussions about the flight plan, head-down time spent programming the FMS, and a determination of the

methods used in programming the FMS with the assigned route clearance.

Results

Demographics

The demographics data showed that the Captains had a mean of 17.2 years of experience as captains. The First Officers had a mean of 11.8 years of experience as first officers. On average, the Captains had 1895 hours of flight time in the 747-400, versus 1880 hours of flight time in the 747-400 for the First Officers; this difference was not statistically significant. Most of the flight crews typically flew long-haul flights between the U.S. and Pacific Rim destinations.

The crews were asked if they had personal computers in their homes, and also if they used computers on a daily basis. These questions were used to assess the crews' familiarity with automation. All of the captains reported having personal computers at home, and 7 of the 10 captains used computers on a daily basis. Nine of the ten first officers had personal computers in their homes and 9 out of 10 reported using computers on a daily basis. This suggests a good deal of familiarity with automation on the part of the flight crews that participated in this study.

Workload Ratings

Across all the simulations, the pilots did not rate their overall workload as excessive, nor were there significant differences in self-reported workload between the different East side fixes, or between the different pilots. In all of the data that were collected, the FMS flight conditions were rated as contributing the greatest amount to their workload, compared to the manual and autopilot conditions. In addition, the autopilot condition was associated with the lowest ratings of workload contribution.

Figure 4 depicts the significant differences found in the self-reported overall workload under the different flight conditions ($F[2,229] = 14.59$, $p < .001$). The average self-reported workload ratings are represented by the heights of the bars. The standard deviations are represented by the "T" lines on the top of each of the bars. The autopilot condition was rated significantly lower than the other two flight conditions ($t[235] = 8.86$, $p < .001$), at a value below the midpoint of the scale. The manual flight condition was rated significantly lower in workload than the FMS condition ($t[235] = -2.92$, $p < .01$).

The pilots were also asked about their satisfaction with their performance (see Figure 5). Overall, they reported mid-level to high ratings of performance satisfaction ($F[2,231] = 12.98$, $p < .001$). In addition, they reported significantly greater satisfaction in performance in the autopilot condition ($t[231] = -4.082$, $p < .001$). The

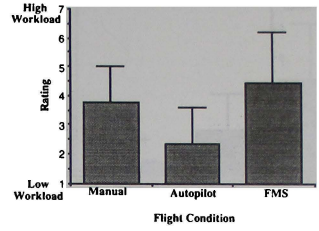


Figure 4. Self-Reported Workload.

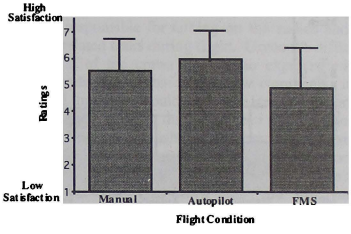


Figure 5. Self-Reported Satisfaction.

FMS mode was also rated significantly lower than the Manual mode in performance satisfaction ($t[231] = 3.078$, $p < .01$).

Figure 6 shows that the FMS mode was associated with the most self-reported frustration levels ($F[2,201] = 49.58$, $p < .001$). This is not surprising, given the lower satisfaction ratings (reported above) that were found in the FMS condition. The frustration ratings were significantly lower for the autopilot condition compared to the FMS and manual conditions ($t[201] = 8.321$, $p < .001$). The FMS condition was rated significantly more frustrating than the manual condition ($t[201] = -5.40$, $p < .001$). Again, these findings mirror the satisfaction ratings by showing that lower satisfaction is associated with higher frustration.

As shown in Figure 7, the activity of planning the flight route and its contribution to the overall workload was also rated significantly differently depending upon the flight mode ($F[2,177] = 21.33$, $p < .001$). The flight planning in the autopilot mode was rated as contributing the least to the overall workload ($t[177] = 4.61$, $p < .001$). The planning activity was rated

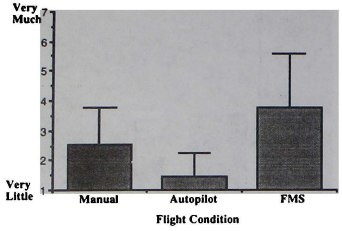


Figure 6. Self-Reported Frustration Level.

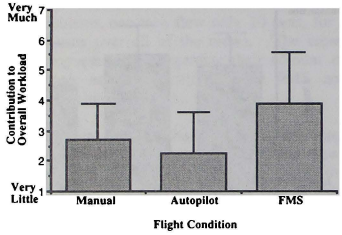


Figure 7. Planning the Flight Route.

significantly different and higher for the manual flight mode, and highest for the FMS mode ($t[177] = -4.58$, $p < .001$). As reported above, in the FMS conditions, where route amendments were issued, these route amendments were the greatest contributor to the overall crew workload.

As shown in Figure 8, time pressure was rated significantly differently depending upon flight condition ($F[2,203] = 38.40$, $p < .001$). The crews rated the time pressure in the autopilot condition as contributing the least to the overall workload, compared to the other two flight conditions ($t[203] = 6.36$, $p < .001$). The FMS condition was again rated higher in time pressure as a contributor to overall workload compared to the manual condition ($t[203] = -5.93$, $p < .001$).

The pilots were also asked about the overall impact of communicating with ATC during the flight scenarios. It would be expected that typical ATC communication would be considered more intrusive as the cockpit tasks increased. However, in the FMS condition, the ATC communication overall is reduced because more

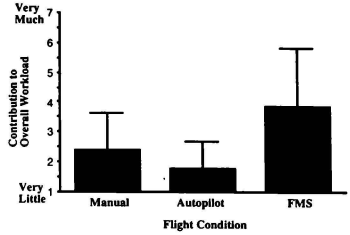


Figure 8. Time Pressure

information on the flight route is provided to the crews via the FMS (rather than by voice). Under all three flight conditions, the crews rated the ATC communication effect on their workload as low to mid-range on the scale (see Figure 9). The crews rated the overall task of communicating with ATC as contributing little to the overall workload under the autopilot condition, followed by being rated as a greater contributor in the manual condition ($F[2,203] = 5.77$, $p < .01$). While the communication with ATC was less demanding under the autopilot condition than the FMS and manual conditions ($t[203] = 2.77$, $p < .01$), the difference between manual and FMS conditions was not statistically significant. Thus, the workload impact of ATC communications under FMS usage was higher than the autopilot "baseline" despite the overall reduction in number of clearances issued by ATC to the crew.

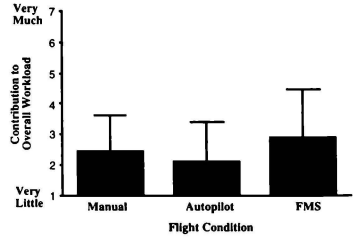


Figure 9. ATC Communication.

Transcript Data

The simulation transcript data was analyzed in three ways: 1) the amount of time devoted to discussions of the flight plan; 2) the amount of head-down time on the part of the pilots; and 3) the different strategies

employed by the crews to implement route changes in the FMS conditions. Results from each of these areas will be discussed in detail below.

Flight Plan Discussion

The activities of a flight crew in the terminal area are traditionally tactical in nature. Crews are provided with instructions from ATC to meet airspace restrictions and are guided to touchdown. In this experiment, the crews were briefed on the flight plan prior to the beginning of the simulations. In addition, under the FMS condition, the planned flight route information was explicitly available to the crew through their route clearances as well as additional approach plates specific to the experiment.

The number of times that a crew discussed the flight plan was tabulated within each of the different flight conditions. Flight plan discussion was characterizedas any discussion regarding upcoming waypoints, future plans for dealing with route amendments, and flight route restrictions. This included discussions when the crews were programming the FMS and discussing the upcoming waypoints and restrictions. Flight plan discussions were not categorized when the crew merely stated their current location along the flight path, or any discussion of their current status in relation to the overall flight path.

As shown in Figure 10, the FMS flight condition resulted in significantly more discussion about the flight plan than the manual and autopilot conditions ($F[2,110] = 56.85$, $p < .001$). There was also greater variability in the amount of discussion about the flight plan in the FMS condition. The manual and autopilot conditions were not significantly different from one another. This suggests the crews discussed the flight plan information more extensively in the FMS condition than in the traditional, more tactical flight environment. This finding also suggests that the crews were more aware of their flight route under the FMS conditions.

Head-downTime

The videotaped data were reviewed to determine the amount of time that the crews were "head-down," or looking at the FMS/CDU to accomplish programming or referencing activities during the flight. The types of head-down activity were classified into 3 categories: Captain (CA) head-down, First Officer (FO) head-down, and both pilots head-down. Over all the flight conditions, the FO head-down category comprised an average of 79.4 seconds of head-down time,followed by the CA head-down category, with an average of 66.4 seconds of head-downtime. There was relatively little overall time in which both crewmembers were categorizedas head-down:29 seconds on average, over all the simulated flights.

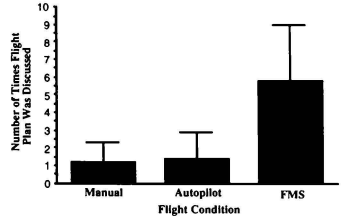


Figure 10. Discussion of Flight Plan.

Head-downtime is necessarily influenced by the flight duties assigned to the PF or PNF. Traditionally, the PNF is responsible for talking on the radios and non-control-related tasks during flight. Given the difficulties that some of the crews appeared to experience in the FMS condition, and based on crew comments, it was expected that there would be some significant sharing of the PNF programming duties between the crewmembers in order to accomplish the necessary programming tasks. Figure 11 depicts the head-down time for both crewmembers in the FMS conditions only. The figure shows that the head-down time did not increase significantly for the PNF. Which crewmember was flying did not significantly influence the amount of head-down time for both crewmembers at the same time. Also, when the CA was flying, the FO did significantly more of the programming and referencing of the FMS ($F[1,87] = 15.64$, $p < .001$). When the FO was flying, the CA did significantly more of the programming and referencing of the FMS ($F[1,85] = 7.77$, $p < .01$). This finding coincides with the traditional duty assignments as requiredby most airline standard operating procedures.

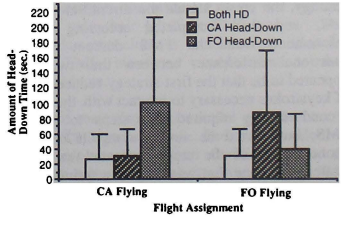


Figure 11. Head-Down Time by Flight Assignment

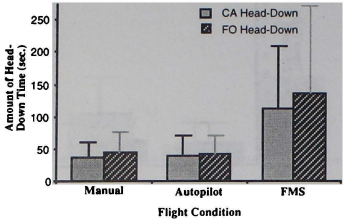


Figure 12. Head-Down Time.

Figure 12 depicts the head-downtime across the flight conditions, partialling out the effects of the PF. There was nearly three times as much head-downtime on the part of the CA under the FMS conditions than the manual and autopilot conditions; this finding is statistically significant ($F[2, 85] = 27.272$, $p < .001$). Similar results were found for the FO head-down time ($F[2,83] = 15.668$, $p < .001$); there was again nearly three times as much head-downtime on the part of the FO in the FMS condition than the manual and autopilot conditions.

FMS Programming Strategies

The FMS programming strategies were examined for the FMS runs. There were five out of the available 46 cases of FMS observations in which no strategy could be determined from the videotaped data.

Two strategies were observed in dealing with the route amendment introduced in the FMS flight conditions. When the crews were provided with the route clearance, they generally followed one of two ways of starting the aircraft along the new route. In the first strategy, the crews got the arrival route information first, before starting the aircraft along its new route. In the second strategy, the crews started the aircraft along the route first, and then completed accessing the arrival information from the FMS descent page. The functional differences between the two strategies appeared to be that the first strategy reduced the number of keystrokes necessary to interact with the FMS. The second strategy required more steps to program the FMS, but started the aircraft along its new clearance sooner. One of the crews indicated that the second strategy was one that was taught in training, and they understood it was the method they were expected to use.

In 63.4% of the available FMS runs, the pilots chose to use the second strategy. Further analysis was conducted to see if there were any statistically significant correlations between the strategy utilized by the flight crews and other experimental conditions (discounting the possible training effects). There were no statistically significant associations between strategy and: the originating fix, amount of head-downtime, the duration of the head-down time, or the PF.

FMS programming time is documented in Reference 3. An analysis was conducted to see if there were significant differences between the programming times associated with the two strategies. The mean programming time was 86.55 sec (SD = 53.35) for the first strategy, while the mean programming time for the second strategy was 117.85 sec (SD = 42.64). Although the perception of the amount of actions required and the mean programming times suggest a definite reduction in programming time for the first strategy, the means were not statistically different from one another.

Pilot Acceptance

Pilot acceptance of the FMS procedures was not explicitly collected with any kind of acceptance rating scale. Acceptance data is therefore inferred from the extensive debriefing discussions held with the pilots following the simulation runs, and at the end of each day of simulation.

Although the crews were largely successful with utilizing the FMS in these terminal-area experimental scenarios, overall, the pilots reported dissatisfaction, and emphasized the extra workload that was required during arrival operations. The additional route clearance change that was issued on runs which originated from the southeastern cornerpost was especially difficult to accommodate. Pilots felt that they had insufficient time to prepare and program for the additional route change. One pilot said that if a change was required late in the approach phase, the change should simply consist of runway headings, speed, and altitude, rather than an entire new route clearance. Another pilot said that when there is low workload, utilizing the FMS reduced the workload even further; however, under higher workload, utilizing the FMS made the overall tasks "impossible."

Discussion

The demographics data show that there was a good deal of familiarity with automation on the part of the flight crews who participated in the experiment. However, this familiarity with automation and the occasional use of the FMS during the cruise portion of long, trans-oceanic flights suggests that the crews' familiarity with programming may not translate into the pressure of utilizing the FMS in a more high-workload environment, such as in a terminal approach segment of flight.

The questionnaire data show that the pilots reported workload to be the greatest in the FMS condition. The crews reported the least amount of workload under the autopilot condition, which could represent a baseline of how they normally perform their flight tasks in the terminal area. One reason for the increased workload in the FMS condition can be attributed to the amount of head-downtime in the terminal area. The time needed to refer to the FMS for programming purposes suggests that there was less time available to perform other required tasks. It was frequently mentioned by the pilots that there is a lot that the crews need to take care of in the final phases of flight, and that monitoring and programming tasks removes a crewmember from being able to participate in these tasks. The impact of ATC communications were expected to be reduced with the use of the FMS in the terminal area, but even though fewer clearances from ATC were issued in the FMS conditions, the impact of communicating with ATC was still considered by the pilots to be higher than the autopilot condition, and about the same as during manual operations.

The amount of time the pilots spent discussing the flight plan was significantly increased in the FMS condition over the other two flight conditions. This result could be attributed to both an increased awareness of the flight plan in general, or indicate problems encountered due to the FMS programming activity. This increase in flight plan discussion was somewhat expected, given that the crewmembers are no longer able to share information in the same way that they would under conventional autopilot operations, such as viewing the setting of switches and dials and associating these motions with intended actions. This increased discussion and awareness of the flight plan might be expected to be associated with greater satisfaction with the overall flight, since more information is being provided to the pilots on what to expect. However, ratings of satisfaction were not positively associated with the FMS conditions. Crews were the least satisfied with their performance in the FMS conditions. In addition, the crews also rated the flight planning as being a greater contributor to the workload under the FMS condition. Reference 3 describes in detail the errors in pilot adherence to trajectories during these simulations: of the nine significant route errors over all the runs, eight were under FMS conditions. The errors consisted of incorrect FMS procedures being entered or the inability to enter the procedures in a timely manner.

From crewmember comments, the head-down time appeared to be a significant issue in the acceptability of the FMS procedures in the terminal area. They perceived that the use of the FMS required too much head-down time, and that it caused both crewmembers to become involved in referencing and programming the FMS regardless of flight condition. Contrary to their perceptions, however, it appeared that the PNF still accomplished the majority of the FMS interactions and the PF was able to devote his attention to the flying tasks. Given the pilot concerns, it might have been expected that both PF and PNF would have experienced more closely equivalent times devoted to referencingand programming the FMS. In contrast, the results showed that the pilots maintained relatively significant differences in head-down time, suggesting that they were able to adhere to standard operating procedures.

The head-down time was much more pronounced in the FMS cases than under the manual or autopilot cases. This suggests that under the more time-critical tasks required of the crews during the terminalapproach phase of flight, utilizing the FMS often means that not only is the PNF spending a great deal of time attending to the FMS, but much more time than is customary under autopilot or manual flight conditions. This may contribute to the increased time pressure that the crews reported under the FMS conditions. The resulting effect may be that under FMS usage in the terminal area, the PF may have to take on more tasks that are usually completed by the PNF. It is critical that further studies examine this issue of sharing of duties. If the FMS programming process is so unwieldy that that PNF must spend too much time programming, the PF will be negatively impacted by having to assist either in traditional PNF flying duties, or in the programming activity itself.

The strategy that the crews employed most frequently in programming the FMS reflects what they have been trained to utilize and may also indicate the more intuitive means of integrating the FMS into the flying duties. In the majority of the cases, the crews chose to use the method which sent the aircraft along its (new) route more quickly, rather than use the quicker way to program the FMS, despite the increase in the amount of keystrokes. Improving the FMS interface through decreasing the amount of keystrokes, or enabling the aircraft to be started along its new route more quickly (with fewer complex FMS programming actions) would probably be a welcome design change. It would also be useful to investigate the different ways that different airlines may train their pilots in FMS usage. This might provide some clues as to airline priorities and how better FMS interfaces would influence their training procedures.

Training itself may also influence the effective use of the FMS in the terminal area. Because the pilots in this study were chosen for their familiarity with the 747-400, they were more likely to have used the FMS only in the cruise phase of trans-oceanic flights. Thus, they may not have had enough familiarity using the FMS features, or identifying and correcting entry errors.

Future studies may wish to provide more training runs to the pilots to familiarize them with the FMS.

Pilot acceptance of the FMS clearances in the terminal area was generally not positive. While the crews were clearly able to accomplish the flying tasks under the FMS conditions, time pressure and crew resource issues contributed to their general dissatisfaction with using the FMS in the terminal approach phase of flight. Other aids may be needed to successfully implement widespread use of the FMS in the terminal area. One such aid is datalink, which could reduce some of the difficulties encountered with programming and entering in route changes that are received by voice. Use of other such devices should consider whether there is an added impact upon head-down time and time pressure experienced by the flight crews.

Conclusion

Previous studies have indicated that the FMS reduces workload, primarily in the cruise phase of flight. In the arrival phase, FMS usage was reported to increase workload.[5] The results of this experiment support these findings. In this study, among the three flight conditions examined, the FMS mode was rated highest in self-reported workload, with the lowest levels of performance satisfaction. The FMS workload levels were rated higher than even flying with minimal automation. Some variables which were assumed would be benefits of using the FMS, such as reduced ATC interaction and possibly more efficient flight planning, were found to actually be more difficult under FMS flight conditions. These expected advantages did not outweigh some of the other problems experienced by the crews.

The self-reported workload under the autopilot condition was the lowest of the three flight conditions. The greater amount of pilot performance satisfaction in the autopilot condition provides an indicator of a baseline level of workload to which pilots are accustomed. This level should probably be considered as a target in the development of cockpit automation and the integration of advanced ATC automation, to help reduce the pilot workload in the terminal area. Strategies observed in the use of the FMS from this study suggest that for future development of FMS interfaces, there should be an examination of what is more intuitive and expedient for the pilots in dealing with route clearances. The time pressure involved in the terminal area is another reason why there should be attention paid to improving the nature of the FMS interface if pilots are to be expected to use the FMS in the terminal area. In addition, training issues should also be addressed in future simulation to help identify whether the terminal area is itself a barrier to FMS usage, or if greater proficiency using the FMS (together with a more intuitive human-computer interface) would be sufficient to allow effective FMS use in the terminal area.

It is clear from this study that while benefits may be expected from FMS usage, further work needs to be done to examine how the pilots can most effectively incorporate this information into their work environment without negatively impacting workload and smooth, safe flight operations. To do so will help to ensure greater accuracy for meeting trajectories introduced by advanced ATC automation. Providing route information to the pilots via datalink could streamline this process in the terminal area, and allow for the efficient use of the FMS.

Acknowledgments

The authors would like to acknowledge the contributions of Diane Lattanzio (Sterling Software, Inc.) and JoAnne D'Souza (Symvionics, Inc.) for their work in the data reduction and analysis of this paper.

References

1. Erzberger, H., "CTAS: Computer Intelligence for Air Traffic Control in the Terminal Area," NASA Technical Memorandum 103959, Moffett Field, CA., 1992.
2. Davis, T.J., Krzeczowski, K.J., & Bergh, C., "The Final Approach Spacing Tool," *Reprints of the 13th IFAC Symposium on Automatic Control in Aerospace, Aerospace Control '94*, edited by D.B. Schaechter & K.R. Lorell, Palo Alto, CA., Sept. 1994, pp. 70-76.
3. Slattery, R.A., Lee, K.K., & Sanford, B.D., "Comparison of Pilot Variations Following Terminal Area Trajectories," To be presented at the *1997 AIAA Guidance, Navigation, and Control Conference*, New Orleans, LA, August, 1997.
4. Williams, C., "Potential Conflicts of Time Sharing the Flight Management System Control Display Unit with Data Link Communications," *Proceedings of the Eighth International Symposium on Aviation Psychology*, Columbus, OH, April, 1995.
5. Chandra, D., "An Evaluation of Automation for Flight Path Management in Transport Category Aircraft," Master's Thesis, MIT, Cambridge, MA., 1989.
6. Sullivan, B.T., Panda, R.C., & Soukup, P.A., "Implementing FANS in the NASA 747-400 Flight Simulator for Airspace Operational Research," *AIAA Flight Simulation Technologies Conference*, San Diego, CA, July, 1996.
7. Sullivan, B.T. & Soukup, P.A., "The NASA 747-400 Flight Simulator: A National Resource for Aviation Safety Research," *AIAA Flight Simulation Technologies Conference*, San Diego, CA, July, 1996.

THREE-DIMENSIONAL VERSUS TWO-DIMENSIONAL DISPLAYS FOR AIR TRAFFIC CONTROL

Mark A. Brown*
STA Research Fellow, National Aerospace Laboratory, Tokyo, Japan
Mel Slater
Department of Computer Science, UCL University of London, United Kingdom

ABSTRACT

An empirical study was carried out comparing two-dimensional and three-dimensional displays for air traffic control, exploring the suitability of 3D displays for ATC. Experiments compared the performance of two subject groups (air traffic controllers and novices with no ATC experience) using three types of display (2D, pseudo-3D and stereoscopic 3D) over a number of tasks: Observations of azimuth angle and distance between two aircraft; selecting two aircraft from a scene based on criteria of highest and lowest, and closest horizontally; and a conflict detection task. The results largely supported the findings of 3D display research, but behavioural differences were observed between the novices and controllers. Evidence was found that subjects tended to detect conflicts by examining horizontal and vertical separations separately rather than exploiting the integrated characteristics of the 3D displays. This may negate some of the benefits of a 3D visualisation, but whether controllers can be trained to exploit fully the integrated presentation of a 3D display remains to be investigation. Finally, controllers invited to view and comment on a virtual reality demonstration rejected immersive VR technology for radar control, but found it potentially useful for visual control room simulation and for workspace design.

INTRODUCTION

Since humans have evolved to make sense of the three-dimensional (3D) visual environment, a 3D display may afford effective communication of spatial information to a viewer 'by exploiting the analytic and integrative characteristics of visual perception'.[1] This gives potentially lower workload and increased efficiency in performing a task involving spatial information compared to a two-dimensional (2D) display, which must use multiple views or multiple codes to convey information along the display's depth axis.

It has been suggested that since air traffic control (ATC) is a task requiring operators to solve problems involving all three spatial dimensions, it is a natural application of 3D display technology,[2] and that the resulting potential workload reduction might enhance safety and efficiency.[3,4] Three-dimensional air traffic displays for ATC and other applications have been implemented in previous studies,[3-7] but despite these investigations, none are currently in operational use. A paramount reason must be the very high risks associated with moving to an unproven technology, especially in a safety-critical application such as ATC. New technologies tend to introduce new problems and errors, the understanding of which tends to lag behind the growth of technology, leading to the danger of misapplication or poor implementation. Any benefits of 3D displays over the *status quo* must be therefore weighed against these and other costs (for example training operators and developing new procedures), but simply casting the an air traffic scenario into a 3D pictorial presentation does not itself guarantee insight into the data.[8]

A key question is therefore how effectively a 3D display can support the task of ATC compared to a 2D display. With this in mind, an study was carried out to compare 2D with 3D displays for ATC.[9] The objectives were to make an exploratory empirical investigation from a human factors viewpoint to identify possible strengths and pitfalls in using 3D displays, and to highlight areas for further investigation. In addition, an immersive virtual reality (VR) simulation was constructed and presented to air traffic controllers for comments as to future applications of the technology.

BACKGROUND

The task of ATC is basically one of flow control and scheduling. Each air traffic control officer (ATCO)* is responsible for a sector of airspace into which aircraft enter at various positions and times and must be guided to their exit points within a

*AIAA member

Copyright © 1997 by Mark A. Brown. Published by the American Institute of Aeronautics and Astronautics, Inc. with permission.

*Air traffic controllers are referred to as Air Traffic Control Officers in the UK, and the abbreviation ATCO will be used henceforth for brevity.

framework of rules and procedures. The task is a four-dimensional problem, with aircraft moving through space and time; flow is managed by looking at the current and projected future state of traffic, and planning accordingly.

ATCOs often refers to their awareness of the air traffic situation as the *picture*, and this forms the key to control. The picture is built and maintained by reference to flight progress strips and a plan-view display (PVD) if air traffic. Studies carried out into the ATCO's picture[10] and computer-supported cooperative working in ATC[11–13] indicate that a large part of the ATCO's task involves working predominantly with the flight strips, with the radar display in support. Conversely, two former ATCOs, Strutt and Burnett, place the emphasis on the PVD. Strutt's assumption is that through the PVD, the controller is constantly aware of the spatial relationships between the aircraft,[3] and Burnett refers to studies which have shown that "Controllers make from 100 to 150 decisions per minute and each decision is based almost entirely on the information presented on the PVD.".[4]

The picture is not so much a pictorial image of the traffic in the mind of the controller as a mental model of the plan, its state of execution and the general disposition of the traffic. It must contain some spatial elements and these elements must have a three-dimensional aspect. However, if there is a spatial model, it may not necessarily be an *integrated* one, possibly due to the difficulties of visualising the vertical dimension from the numeric height information presented on the PVD and flight strips. This raises two questions: (1) If the ATCO's picture is not an integrated 3D spatial image, then would a 3D traffic image support the air traffic control task? (2) What can a 3D air traffic display offer that might assist the formation and maintenance of the controller's picture?

In making the choice between 2D and 3D display formats for an application, Wickens and Todd suggest two research domains which should be considered: 3D display research and the proximity compatibility principle.[1]

The proximity compatibility principle asserts that "tasks of a more integrative nature involving (for example) the comparison between data points will benefit from more 'object-like' displays, whereas tasks requiring the focus of attention on a single dimension or single object will be better served by more separated bargraph or digital displays."[1] Applying this to ATC, which display is better suited depends on how the spatial separations between objects are compared and assessed, whether as separate horizontal and vertical dimensions or in an integrated manner.

Regarding 3D display research, although the 3D picture presents spatial information in a format which is easily interpreted by the viewer, there are a number of drawbacks, the most critical of which are that the 3D representation creates ambiguity regarding the precise location of objects along the display's depth axis (and consequently potentially reduced precision in reading values along any one particular axis), and that the multiple parameters involved in constructing the display interact in complex ways and can lead to distortions or biases in viewing.[2,14] If ATC tasks require spatial judgements such as distance and angle between objects to be made with precision, then a 3D rendering might not be suitable.

Some Previous Research

Tam[7] and Strutt[3] have constructed concept demonstrations of 3D air traffic representations for ATC.

Strutt implemented a display comprising three orthogonal 2D views (one plan, two side) and an integrated 3D view, which he suggests might fix the picture in the controller's mind and alleviate mental integration workload. He postulates two reasons for the non-operational implementation of 3D displays in ATC to date: (1) A plan-view display can already convey 3D information through the digital height readout in the datablock, and (2) previous 3D representations may have caused controllers to become disorientated, perhaps due to position ambiguity caused by lack of depth information.

Tam implemented a 3D display for visualising a new ATC concept known as 'tubes of flight', in which aircraft are assigned tunnels in the sky. His display features a PVD with a sub-window in which a selected portion of the airspace can be rendered in 3D and manipulated (rotated, scaled etc.) independently of the main display.

The studies of Tam and Strutt did not experimentally compare the 3D and 2D formats. Two studies which do so are those of Burnett & Barfield[4,15] and Bemis, Leeds & Wiener.[6]

Burnett & Barfield's study used a survey and experiments involving ATCOs to analyse the effects of multiple colours, information density and traffic complexity on 2D and perspective 3D displays. They measured the times ATCOs took to perform three tasks: Resolution of impending conflicts, identification of highest and lowest aircraft in a scenario, and the reconstruction of air traffic scenarios. It was found that performance using a perspective display was at least equal to and in some cases better than the PVD. Subjects ex-

pressed a preference for the 3D display for extracting immediate spatial and directional information, but indicated that due to their training on PVDs, they were almost entirely dependent on datablocks for speed and altitude extraction.

Bemis *et al.* compared plan-view and perspective tactical displays for the tasks of threat detection and interceptor selection, and found that subjects using the perspective display made significantly fewer errors and took less time to perform the threat detection and interception tasks.

THE STUDY

The object of the study was to compare 2D and 3D display formats over a series of tasks which were felt to be pertinent to ATC, based on verifying and extending extending the study of Burnett & Barfield.

Experiment Overview

In order to address the problem of bias due to prior training and experience, two volunteer subject populations were used: 22 ATCOs[†] (referred to as the 'expert' group), and 16 subjects with no ATC experience (referred to as the 'novice' group)[‡]. The ATCOs' ages were distributed fairly evenly between the ranges 21–30, 31–40 and 41–50 years, whereas the novices were predominantly in the 21–30 years age group. All but two of the novices and four of the ATCOs were male. It was hypothesised that the ATCO group might have a higher average spatial ability, and that this might explain any differences in performance between the groups; however a spatial ability test[16] failed to find any significant difference between the spatial abilities of the two groups.

The experiment was conducted as a 2×3 factorial design (two levels of subject group crossed with three levels of display type (PVD, pseudo-3D and stereoscopic 3D, described below). Subject cell allocations are shown in Table 1. The distribution is uneven due to 'no shows' and rescheduling of subject attendance.

	Display			
Group	PVD	pseudo-3D	stereo-3D	Total
Novice	6	5	5	16
Expert	7	9	6	22
Total	13	14	11	38

Table 1 Number of Subjects in Each Cell

[†]Due to recruitment restrictions, all participating ATCOs worked at visual control room tasks, with only a few current radar controllers. However, all except one had previous operational radar control experience, and all had received radar training.

[‡]The novice group was composed of 13 university students and 3 lecturers/teachers.

There were several experimental tasks. Two of these concentrated on elements of the ATC task in isolation: Interpreting angles and distances; and speed and accuracy of selection of aircraft based on criteria of horizontal proximity and altitude. A conflict detection task was included as compound task to apply the isolated elements of the first two tasks to a more complex 'real-world' one. In addition, an immersive VR system was used to present a visualisation of air traffic and an architectural simulation to the ATCO group, in order to canvass subjective views regarding the utility of VR technologies for future ATC applications.

Following a demographic questionnaire, each subject completed the tasks sequentially, with a training session prior to and a questionnaire following each task. Rest time was given between each task. The total time taken to perform all tasks was not recorded, but no subject took more than 3 hours to complete the experiment (including the 20 minute spatial ability test).

Displays

Three types of display were developed for the study: A 2D plan-view display (PVD); a pseudo-3D perspective display, using only monocular cues to convey the sense of depth; and a stereoscopic 3D display. A schematic of the pseudo-3D display is shown in Figure 3. (Note that this is a monochrome reproduction, whereas the original displays used colour, as per an interim NATS colour standard.[17]) The displays showed a 50×50 n.mi. area centered at the radar head at Heathrow airport, with a video map showing airway structures in the London Terminal Manœuvering Area. Range rings were superimposed at 10 n.mi. intervals as an aid to distance judgement. North was in the direction of the display depth axis.

Two types of projection were considered for generating the 3D display: a perspective projection, which contains the depth cue of linear perspective, and a parallel projection, which does not. An example of the prototype perspective 3D display format is shown in Figure 4. The parallel projection was selected after evaluation in a pilot study. Reasons included the fact that linear perspective led to 'clustering' of aircraft close to the horizon, and meant that the projected size of a vector of fixed objective length varied depending on its depth in the display.

The displays were implemented on a Silicon Graphics Indy workstation, displayed on a 20-inch colour monitor with a dot pitch of 0.26 mm. The stereoscopic display was implemented using time-multiplexed images—left and right fields were

presented alternately at a rate of 60 Hz and the subject wore Stereographics CrystalEyes liquid-crystal shutter glasses synchronised to the display to blank the left or right eye as necessary. In this particular implementation, display vertical resolution was halved when the display was in stereoscopic mode (the number of scanlines was halved, but the picture expanded vertically to fill the screen).

The amount of aircraft symbology and the precise datablock content varied with the task (the symbology is shown more clearly in Figure 4). Aircraft position was represented on the PVD by a black filled circle of 1 mm diameter, and could be followed by smaller trailing histories in the case of a dynamic scenario. The 3D displays used two sets of position symbols, for the air and ground positions, with the ground symbols rendered in white. The air plot and ground plot were connected by a 'drop line', the length of which gave a direct visualisation of the height of the aircraft.

The datablocks were rendered as black text on a solid white background with a border, at the 12 o'clock position with respect to the aircraft position symbol. In the 2D display, overlapping datablocks could be 'swapped' by clicking with a mouse, but it was found that with this approach in the 3D display, 'far' datablocks could obscure 'near' datablocks, violating the occlusion depth cue and partly destroying the sense of depth. In the 3D displays, therefore, clicking on a datablock which obscured another caused it to become transparent, just leaving its frame and revealing the datablock behind (see Figure 3).

The tasks and their results are now presented in brief. Lack of space precludes reporting the statistical analyses in detail: Readers are referred to Brown(1996)[9] for full information.

Task 1: Azimuth Angle & Relative Distance

In informal interviews conducted with ATCOs prior to the experimental study, the ability to make sufficiently accurate judgements of azimuth angle and relative distance between aircraft was stated as being of great importance. This task was designed to test the influence of display type and subject group on observations of these parameters. Twenty stimuli each consisting of two co-altitude aircraft (randomly placed subject to a maximum distance constraint) were constructed and were presented to the subjects in random order. For each stimulus, the subjects were asked to note the angle and distance between the two aircraft on a response sheet, working at their own pace.

Due to the biases in reading azimuth and elevation angles[14,18,19] and problems of distance ambiguity in 3D displays, it was anticipated that angles and distances would be read more accurately using the 2D display than either of the 3D formats. The stereo-3D display was expected to afford greater accuracy than the pseudo-3D display, since stereo viewing removes one of the sources of viewpoint misestimation responsible for exocentric azimuth judgement biases in perspective displays.[14] These hypotheses appear to be supported by the results. Analysing the azimuth angle observations by an analysis of covariance, little overall bias was found in the case of the PVD (< 1°), whereas subjects using the 3D displays showed a tendency to underread angles, the bias being larger for the pseudo-3D display (-2.8°) than for the stereo-3D display (-2.4°). Similar results were found when analysing the distance observations[§]—the 2D display gave the highest accuracy, followed by the stereo-3D display, with the pseudo-3D display affording the lowest accuracy.

Due to their experience and training, ATCOs were expected to read angles and distances with greater precision than novices. However, it was found that subject group did *not* have a significant effect on observations of either azimuth angle or distance. This may have been due to behavioural differences between the two groups. It was observed that novices tended to take more time over the task, possibly in an attempt to achieve high accuracy, whereas the ATCOs may have been satisfied with attempting to read to a lower level of accuracy. In support of this hypothesis, it was found that the experts had a higher proportion of azimuth angle observations which were multiples of 5° than novices (97% and 88% respectively), and for distance, observations which were whole numbers formed a higher proportion of expert observations than those of the novices (90% and 75% respectively).

Task 2: Information Extraction

This task investigated how the speed of extracting horizontal and vertical information depended on display format and subject group.

A pilot study found that ATCOs filter nonpertinent traffic (e.g. traffic above or beneath the sector), and also tend to consider the horizontal and vertical dimensions separately for the purposes of separation. Burnett also states that the primary interests to controllers are vertical and

[§] Analysis of covariance had to be applied to the logs of the actual and observed distances, due to non-constancy of variance of observed distance with stimulus distance (the variance of observations got larger the further the targets were apart).

lateral separations between aircraft, and that a 3D display would allow controllers to 'see and interpret' these. Further, she suggests that certain types of information might be interpreted more quickly using a 3D display than a PVD (e.g. the identification of an object as high or low, near or far, ascending or descending),[4] and she found that subjects were able to identify the highest and lowest aircraft faster using a 3D display than a 2D display.[15]

The information extraction task was divided into two parts. The first part required subjects to select the highest and lowest pair of aircraft from a number; the second part required the identification of the two aircraft with the smallest lateral separation. For each part of the task, subjects were presented with 10 stimuli each containing 10 aircraft, and for each stimulus were required to select two aircraft according to the prevailing criteria using a 3-button mouse. Subjects were instructed to work as quickly as possible whilst maintaining accuracy.

The time from presentation of each stimulus to the selection of the two aircraft was recorded automatically. The aircraft were selected by 'clicking' on their position symbols (the air position symbol only for the 3D display formats) with the left mouse button, which toggled the symbol between selected (red) and non-selected (black). When satisfied with the selections, the subjects were required to press the middle mouse button, which caused the time elapsed since the start of the stimulus to be recorded and the next stimulus to be presented. A number of problems were found with this mechanism, however: Some subjects initially forgot that only the *air plot* symbols were sensitive on the 3D display; the mouse was sometimes 'sticky' (despite regular cleaning); and there were occasional problems with subjects accustomed to single-button mice pressing the wrong button. These problems led to a few data points with times markedly greater than the rest, but the technique of leverage plots[20] showed that these were below the threshold at which they could be considered as outliers, so analysis proceeded using all the data collected.

Altitude Extraction

It was anticipated that subjects using the 3D displays would be able to identify and select the highest and lowest aircraft faster than those using the PVD, since determining height on the PVD requires all the datablocks to be read, whereas in a 3D display the heights of aircraft are visualised graphically. Subject comments suggested that in the 3D display formats, the length of the drop lines were used to narrow down the search, with the datablocks used for confirmation and accuracy. However, it was also pointed out that this strategy was not useful where drop lines were of similar length, and some subjects also reported problems in comparing drop lines of similar lengths which were far apart on the display.

Descriptive statistics of the altitude extraction times are given in Table 2. Analysis of variance

Display	Novice	Expert
PVD	11.2 ± 4.5, 60	12.7 ± 4.4, 70
pseudo-3D	14.2 ± 8.1, 50	11.5 ± 4.3, 90
stereo-3D	9.6 ± 2.3, 50	12.0 ± 5.0, 60

Table 2 Altitude Extraction Times: $(\overline{X}(\text{sec}) \pm s, N)$

(ANOVA) tests suggest that display type had a significant influence for the novice group: the display ranking in descending order of altitude extraction performance was stereo-3D display, PVD, pseudo-3D display. The fact that the performance of subjects using the PVD was not the slowest was unexpected, and may be due to the problems with clutter and inadequate depth cues making the pseudo-3D display difficult to interpret. It is postulated that with adequate depth cues (in this case, with stereopsis), a 3D display of aircraft position affords faster performance than a 2D display for determination of height, but if depth cues are not adequate, then the potential benefits of the 3D presentation may be negated.

For the expert group, display type was found to have no significant influence. This might be explained by a bias towards the use of datablocks: Some ATCOs reported a tendency to use the datablock height information even in the 3D representations, citing included force of habit and doubts about the accuracy of using the drop lines as reasons.

Using the PVD, the novice group was found to be faster than the ATCO group (although this was borderline significant—$p = 0.052$). For the stereo-3D display, novices were significantly faster than the experts. No significant difference between the groups was observed in extraction times for the pseudo-3D display. Two possible reasons for the observed differences in performance between the two groups are: (1) That the novice group might be more experienced in using mice and so less prone to selection errors, and (2) differences in the way in which information was extracted from the display—expert subjects may have been more concerned with accuracy (which would tend to place an emphasis on datablock information) rather than speed (which would tend to emphasise the use of the drop-lines in the 3D display formats), whereas the novice group could have been more concerned with speed than accuracy.

Horizontal Proximity Extraction

The PVD presents horizontal distance information unambiguously, whereas the 3D display is subject to distance ambiguity. It was hypothesised that subjects might be slower at extracting horizontal separation information from a 3D presentation than a 2D presentation.

Descriptive statistics of the horizontal proximity extraction times are given in Table 3. A two-factor ANOVA analysis suggests that display has a significant influence over horizontal proximity extraction time. The ranking of displays in order of decreasing performance was: PVD (fastest), stereo-3D, pseudo-3D (slowest). Subject group was not found to be significant.

Display	Novice	Expert
PVD	6.2 ± 3.7, 60	6.9 ± 2.6, 70
pseudo-3D	11.7 ± 8.7, 50	10.2 ± 5.2, 90
stereo-3D	8.9 ± 5.4, 50	9.1 ± 5.9, 60

Table 3 Horizontal Extraction Times: $(\overline{X}(\text{sec}) \pm s, N)$

Some of the selections were incorrect, and in some of the stimuli, only the subjects using a 3D display made incorrect selections. These stimuli contained more than one pair of aircraft for which horizontal separations were very similar, but one of the pair lay along a direction roughly coplanar with the display depth axis while the other pair lay along an direction roughly orthogonal with the depth axis. The fact these stimuli gave incorrect selections only in the 3D display formats is therefore probably due to the ambiguity of position in the 3D displays.

Some subjects commented that they found the elliptical range rings in the 3D projections difficult to use, which may have contributed to the poor performance found using the 3D displays. Further, there may have been an additional search time for the 3D displays since the instructions stipulated that the subject had to select the aircraft *air* plots, but closest horizontal proximity must be determined on the *ground* plots.

Task 3: Conflict Detection

Although ATC is carried out by using a combination of flight strips and radar, the investigation of the role of the flight strips is beyond the scope of this study. This task concentrates on whether or not a 3D radar presentation is more or less effective than a PVD for detection of potential conflicts. It was postulated that because of its integrated presentation of the three spatial dimensions and reduced mental integration workload, conflict detection performance (in terms of time to detect a conflict and correct identification of conflicting aircraft) would be greater using the 3D display formats than with the 2D display.

The task of conflict detection, including the concept of separation, had to be explained in sufficient detail for the novice subjects to be able to carry out the task. However, there were limits as to the amount of training which could be feasibly given in the time available. For the benefit of controllers, some route information was incorporated into the aircraft's datablock. However, it was recognised that this information would be useless to the novices.

Subjects were presented with two training scenarios followed by six experiment scenarios in random order. Each was a dynamic traffic scenario comprising 90 s of animation (one frame every 6 seconds) and contained either a conflict or no conflict (two no conflict, four conflict scenarios, including a difficult-to-spot 'altitude bust'). Subjects had no *a priori* knowledge of whether or not the scenario contained a conflict. A blank screen was presented prior to each scenario, which was initiated by the subject. If they perceived no conflict, subjects were required merely to let the scenario run to completion. If a conflict was detected, subjects were required to stop the scenario by pressing a button, then to select the two conflicting aircraft with a mouse in the same manner as for the altitude/horizontal proximity extraction task. After all scenarios were completed, the subjects were asked to describe how they carried out the task.

Conflict Detection Times

A two-factor ANOVA analysis of conflict detection times found that subject group had a significant influence, the expert group (41.2 ± 20.6 s, $n = 74$) being significantly faster than the novice group (49.1 ± 18.5 s, $n = 43$), but the display did not and there was no significant interaction between display and subject group.

Conflict Detection Responses

Conflict detection is a difficult task to assess because it is partly subjective. As Burnett & Barfield note, "... because each controller develops a different traffic management style, one controller may perceive a conflict exists for any given situation and a different controller may perceive that a conflict is non-existent for the same situation if certain procedures are immediately executed".[15] Some subjects commented that not being actively in control and having the overall plan made the task more difficult, as they were unsure of the intentions of the aircraft. They may have thus preferred to take action on a 'better safe than sorry' basis. It is therefore a slight misnomer to classify some

responses as 'incorrect' in cases where individuals perceived a conflict to exist where there wasn't one, but it is informative to analyse the results on that basis.

If a scenario contained a conflict and the subject selected that conflict, or if the scenario contained no conflict and the subject indicated that there was no conflict, this was classified as a *correct* response. A response was *incorrect* where subjects either failed to identify the correct conflict in scenarios containing a conflict, or indicated a conflict in scenarios which did not contain a conflict. Analysing the percentage of incorrect responses for each cell by logistic regression, it was found that subject response depended significantly on the display type but not on group, and there was no interaction effect between display and group. The proportion of incorrect selections per display type were: 2D PPI 12.8% (10 of 78), pseudo-3D 31.0% (26 of 84), stereo-3D 24.2% (16 of 66). Thus, significantly more incorrect selections occur for subjects using the 3D display formats than for subjects using the PVD.

One scenario in particular had significantly more 'incorrect' conflicts detected for the 3D displays than for the 2D display—two aircraft lying approximately in the plane of the display depth axis were falsely perceived to be in conflict. It is speculated that the situation was less clear with the 3D display due to distance ambiguity.

Many subjects failed to identify the altitude bust conflict (where a climbing aircraft failed to level off at its cleared altitude, but continued to climb), but there was no significant effect of subject group on the number of incorrect responses in this scenario, suggesting that no type of display format favoured detection of this particular class of conflict.

A problem encountered was excessive clutter due to datablocks where a number of aircraft were in close proximity. This particularly appeared to be a problem in one scenario, where there were several aircraft around the location of the conflict and where all the incorrect responses were cases of the conflict not being identified. The problem appeared to be worse in the 3D display cases than in the PVD because of the greater amount of symbology in the former.

Conflict Detection Techniques

These stated techniques used by novice and expert subjects were found to be remarkably similar, except that novices tended to disregard aircraft speed. The overall technique seems to be as follows: (1) First, scan for immediate conflicts by looking at ground position (for lateral separation) and height (for vertical separation); (2) Eliminate aircraft unlikely to be in immediate confliction. For the other aircraft, look at manœuvres tracks (as the trailing histories appear)¶ and vertical manœuvres—to look for impending conflictions: (3) Continue scanning, with higher priority to some aircraft than others. For the 3D display formats, where subjects reported the type of position information they used, ground position was mentioned far more than air position.

Discussion

Surprisingly, the behaviour of the expert and novice groups was found to be very similar. The only significant difference in quantitative results was that experts tended to identify conflicts sooner than novices. There was no evidence that the 3D display aided conflict detection—there were significantly more 'incorrect' than 'correct' responses for the 3D display formats than for the PVD, and no significant influences of display type on conflict detection time were found. This is contrary to the findings of Burnett & Barfield, and possible reasons for the failure of 3D displays to show any significant performance benefits over the PVD include:

1. The inherent ambiguity of position along the display depth axis in a 3D display may lead to problems with detecting conflicts between two aircraft where one or both are travelling approximately in the plane of depth axis.

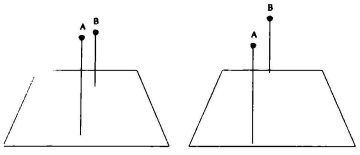
2. Excessive clutter. Subjects reported aircraft-related symbology being 'garbled' against the datablocks, and that excessive datablock overlap increased workload. Since the 3D display formats contained twice as much aircraft-related symbology as the PVD format, they were more prone to this problem. The display scale was also excessively wide, leading to crowding in areas of activity.

3. There is evidence that subjects considered vertical and horizontal information separately when assessing conflicts, and when referring to horizontal position, they referred to ground position. The air position symbols would not therefore have been of much use.

4. Inadequate depth cues leading to difficulties in determining position. Figure 1 illustrates two such problems encountered.

A future study should perhaps repeat this experiment, but with an improved design of display less prone to clutter.

¶When the scenarios started, aircraft did not have trailing histories—these built up as the scenario unfolded, so track information could not be immediately assessed.



(a) Ambiguity of Position
Air plots of A and B may be close whilst their ground positions may be far apart.

(b) Height in the Visual Field as a nuisance cue.
Failure to consider ground position may result in B being perceived as at a greater altitude than A.

Fig. 1 Position Ambiguity in 3D Display

Virtual Reality Demonstration

An informal assessment was carried out of whether currently-emerging VR technology might be applied to ATC. Following the experimental tasks, expert subjects were given the opportunity to experience two simulation scenarios and were invited to comment. The scenarios were a dynamic air traffic visualisation and, incongrously, a kitchen interior. (The latter was not originally planned, but an early subject commented that a new visual control room was in the process of being designed, and VR seemed a natural application for interior design.)

The visualisations were implemented on a Division ProVision 100VTX machine, with a hand-held Division 3D mouse with pushbuttons used for navigation and the scene presented through a Virtual Research Flight Helmet head-mounted display (HMD), shown in Figure 2. The Flight Helmet is a binocular display with a horizontal field-of-view of 75° and uses two colour LCDs of 360×240 pixel resolution to present the stereoscopic images. The degree of binocular overlap in not easily adjustable, and the interocular distance is fixed.

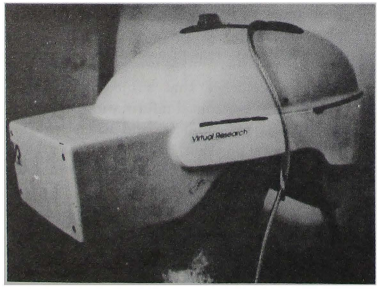


Fig. 2 Head-Mounted Display

Regarding the display system itself, subjects commented on technical shortcomings: Weight of the HMD, poor resolution, slow update rate, and difficulties with navigating in the virtual environment. For operational application in ATC, these comments indicate that current technology would have to be a lot more mature than at present.

Subjects universally rejected immersive displays were for radar control work, despite evidence that they found the sensation of depth greater than in either of the 3D monitor-based displays used in the experiments. Reasons for the rejection of the display included:

- An exocentric viewpoint, as opposed to the egocentric viewpoint of immersive VR, is required such that the controller is outside the scene and able to see everything, otherwise there is the danger of missing something outside the field-of-view.
- One controller stated the need for a fixed image against which to assess location and movement for conflict detection. Another thought the changing viewpoint to be potentially disadvantageous.
- Potential for disorientation.
- No obvious benefits over present equipment.
- The physical and physiological effects of prolonged use of immersive equipment are not known.
- ATC is a highly cooperative system, where controllers have to operate as a team, so there is a need to cooperate with other controllers rather than to be encased in one's own 'world'.
- The controller would still require access the various tabular displays (e.g. flight strips, maps, weather reports) and controls (e.g. R/T channel selection), and these would have to be provided in an immersive environment.

Responses to the kitchen visualisation were substantially more positive, although with reservations. Subjects mostly saw applications in VCR simulation for limited training and potentially for the design of new VCRs, especially assessing lines of sight. For training, limitations of VR compared to other VCR simulation and training aids include the lack of ability to write on flight strips, poor display resolution and cumbersome technology, but VR has the potential to be more cost-effective and flexible than a full VCR simulator in some part-task training applications, if not for full-task simulation. For new VCR design applications, controllers liked the possibility of using a VR simulation to assess lines of sight from the operator's positions to different parts of the aerodrome (which was raised many times as an area

of concern), layout of furniture, access etc. These might be easier to assess in an immersive visualisation than in a monitor-based simulation, enabling effective prototyping and development.

CONCLUSIONS

To recapitulate, the aim of this study was to explore possible issues in adopting 3D displays for ATC, comparing with the current 2D *status quo* in an attempt to elicit the relative strengths and weaknesses of each. This was accomplished by comparing empirically the performance of subjects performing tasks using different types of display. The intention was not to perform each task as much in-depth as a full investigation of each specific task area would require—rather, each task was designed as a preliminary exploration within the framework of the aforementioned objectives.

The results largely supported the findings of 3D display research, but the behavioural differences between the expert and novice subject populations were sometimes unanticipated. For example, contrary to expectation, no significant differences were found in the accuracy of azimuth angle or distance judgements between ATCOs and university students, even on the 3D displays, and ATCOs were reluctant to exploit the 3D characteristics of the display when selecting the highest and lowest pair of aircraft in a scenario, preferring to use the alphanumeric datablocks, citing accuracy and familiarity, which lead to significantly slower performance than novices using the same display. This indicates that simply presenting subjects with new display features does not guarantee that those features will be used effectively, reinforcing the need for additional training for operators using 3D displays.

As anticipated, subjects estimated azimuth angles and distances less accurately using the 3D representations than the PVD. The magnitude of the error depends in part on the specific parameters used to generate the display, and these may be optimised to reduce the error; however, since there are several interacting parameters this may not be an easy task. A number of measures have been suggested for reducing error, such as training to compensate for display distortion, and symbolic enhancements.[14] Before error-reduction studies can be carried out, however, a more fundamental question needs to be answered: What is the magnitude of error which can be tolerated in various radar control tasks? In answering this question, it should be borne in mind that some applications may be more critical than others; for example, issuing radar vectors to place inbound aircraft on an extended runway centreline at an approach control position may require more precise estimation of azimuth angle than en-route control.

Burnett suggested that a 3D display might enable horizontal and vertical separations to be perceived more clearly, and that certain types of information might be interpreted more quickly using a 3D display than a PVD. This research only partially supports this, although it is recognised that problems with implementation may be partly responsible. However, some salient points and issues were raised.

Regarding the height extraction task, the drop line in this implementation did not actually allow height to be determined with precision. The authors suggest that whether or not aircraft can be quickly identified as 'high' or 'low' in a 3D display is of limited benefit to the ATCO, and it is the visualisation of vertical separation rather than actual height which is of interest. However, the value of the 3D display may be negated if the vertical separations between aircraft at similar altitudes cannot easily be perceived, forcing the operator to rely on the datablock. This suggests that only a restricted band of heights should be displayed to ensure that vertical separation is always clearly visible, or that other cues (such as colour) or scaling should be employed (however, unequal vertical and horizontal scales could lead to other difficulties). Obviously, the problem of how to represent an aircraft of unknown height (e.g. general aviation traffic not equipped with mode-S transponders) also remains.

Regarding horizontal separation, it is recognised that speed of at which subjects can perform the task of selecting the aircraft in closest horizontal proximity may itself be of little practical relevance. However, this task and the conflict detection task found evidence that the ambiguity of position along the display depth axis might cause problems with determining horizontal separations, in that two aircraft in the plane of the depth axis might be perceived as closer together than is actually the case.

The study found that in carrying out the conflict detection task, both groups of subjects showed evidence of thinking in horizontal and vertical dimensions separately, rather than in an integrated fashion. Invoking the proximity compatibility principle, this raises questions as to the utility of using an integrated 3D display for a non-integrated task, particularly since the non-integrated display can convey horizontal and vertical information more accurately. However, the way in which the task was carried out may have been due to the fact that ATC has evolved with 2D displays, and possibly to do with the limitations of the 3D implementations in this study. A future investigation might exam-

ine whether the integrated characteristics of a 3D display may be exploited for conflict detection.

Problems were found with the 3D presentations which are potential problems for any such visualisation. These include increased clutter due to the greater amount of symbology present than in the 2D display, choice of display parameters (too great a horizontal and vertical range make separations between aircraft in close proximity difficult to see) and inadequate depth cues.

Finally, while evidence from this type of research should be considered, it cannot alone predict how operators will perform using the displays for a complex task such as ATC. An evaluation of 3D displays for ATC should therefore use a full operational simulation with conditions as close to reality as possible. Only with such an evaluation can all the problems and issues be identified.

*

References

[1] Wickens, C. D. and Todd, S., "Three-Dimensional Display Technology for Aerospace and Visualisation," *Proc. Human Factors Soc. 34th Annual Meeting*, 1990, pp. 1479–1483.

[2] Wickens, C. D., Todd, S., and Seidler, K., "Three-Dimensional Displays: Perception, Implementation and Applications," Tech. rep., Aviation Research Laboratory, Institute of Aviation, University of Illinois at Urbana-Champaign, 1989, Subcontract for AAMRL/HEL/CSERIAC Wright-Patterson Air Force Base, Contract DLA900-88-D-0393.

[3] Strutt, G. G., *The Human Computer Interface in Air Traffic Control: Feasibilty study for a Three Dimensional Radar Picture*, Master's thesis, Middlesex Polytechnic, London, UK, 1991.

[4] Burnett, M. S., *An Analysis of the Effects of Multiple Colors, Information Density, and Traffic Complexity on Plan View and Perspective Air Traffic Control Radar Displays*, Master's thesis, Dept. of Industrial Engineering, Univ. of Washington, 1991.

[5] Ellis, S. R., McGreevy, M. W., and Hitchcock, R. J., "Influence of a Perspective Cockpit Traffic Display Format on Pilot Avoidance Maneuvers," *AGARD Conference Proceedings*, 1984, pp. 16-1–16-9.

[6] Bemis, S. V., Leeds, J. L., and Wiener, E. A., "Operator performance as a function of type of display: Conventional versus perspective," *Human Factors*, Vol. 30, No. 2, 1988, pp. 163–169.

[7] Tam, B., *Three Dimensional Display for Air Traffic Control*, Master's thesis, Dept. of Computer Science, Univ. of Manchester, Mar. 1993.

[8] Ellis, S. R., "Vision of Visualization Aids: Design Philosophy and Observations," *SPIE*, Vol. 1083, 1989, pp. 220–227.

[9] Brown, M. A., *An Investigation of Three-Dimensional Displays or Real-Time Safety-Critical Command Control Applications: with application to Air Traffic Control.*, Ph.D. thesis, Dept. of Computer Science, QMW College, University of London, 1996.

[10] Whitfield, D. and Jackson, A., "The Air Traffic Controller's "Picture" as an Example of a Mental Model," *Proceedings of the IFAC/IFIP/IFORS/IEA Conference on Analysis, Design and Evaluation of Man-Machine Systems*, Baden-Baden, 1982.

[11] Hughes, J. A., Randall, D., and Shapiro, D., "Faltering from Ethnography to Design," *CSCW '92 Proceedings*, ACM, Nov. 1992, pp. 115–122.

[12] Harper, R. H. R. and Hughes, J. A., "What a F-ing System! Send 'em all to the same place and then expect us to stop them hitting: Making technology work in air traffic control," *Technology in Working Order: Studies of work, interaction, and technology*, edited by G. Button, chap. 7, Routlege, 1993, pp. 127–144.

[13] Shapiro, D., Hughes, J. A., Randall, D., and Harper, R., "Visual re-representation of database information: the flight data strip in air traffic control," *Cognitive Aspects of Visual Languages and Visual Interfaces*, edited by M. J. Tauber, D. E. Mahling, and F. Arefi, chap. 4, Elsevier Science, 1994, pp. 349–376.

[14] Ellis, S. R. and Hacisalihzade, S. S., "Symbolic Enhancement of Perspective Displays," *Proc. Human Factors Soc. 34th Annual Meeting*, 1990, pp. 1465–1469.

[15] Burnett, M. S. and Barfield, W., "Perspective versus Plan View Air Traffic Control Displays: Survey and Empirical Results," *Proc. Human Factors Soc. 35th Annual Meeting*, 1991, pp. 87–91.

[16] Smith, P. and Whetton, C., *General Ability Tests*, ASE, A Division of NFER-NELSON, 1988.

[17] Reynolds, L. and Metcalfe, C., "The Interim NATS Standard for the use of Colour on Air Traffic Control Displays," CS Report 9213, Issue 1.2, UK CAA, Chief Scientist's Division, National Air Traffic Services, 1992.

[18] Ellis, S. R., Tharp, G. K., Grunwald, A. J., and Smith, S., "Exocentric Judgements in Real Environments and Stereoscopic Displays," *Proc. Human Factors Soc. 35th Annual Meeting*, 1991, pp. 1442–1473.

[19] McGreevy, M. W. and Ellis, S. R., "The effect of perspective geometry on judged direction in spatial information instruments," *Human Factors*, Vol. 28, 1986, pp. 439–456.

[20] Howell, D. C., *Statistical Methods for Psychology*, Duxbury, 3rd ed., 1992, ISBN 0-534-92955-9.

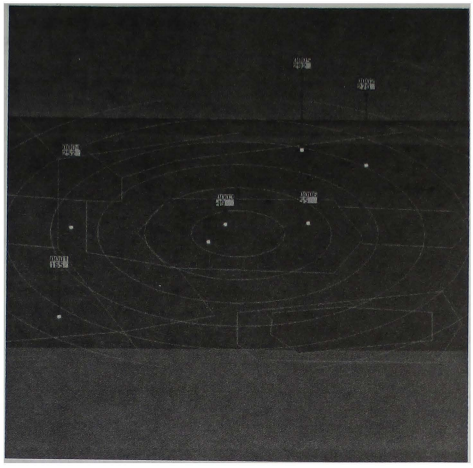


Fig. 3 Experiment 3D Display Format

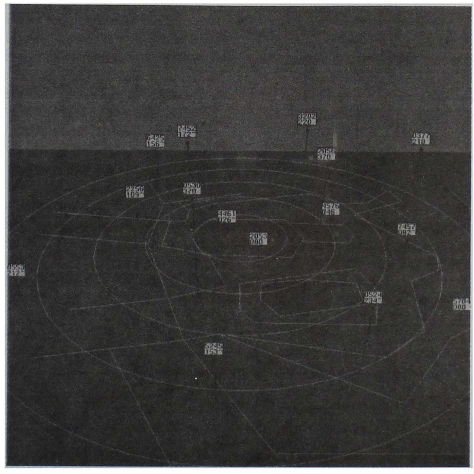


Fig. 4 Prototype Perspective Projection 3D Display format

Control and and Dynamics Simulation Facility at Hughes Space and Communications

Steve Zammit
Hughes Space and Communications Company
El Segundo, California

Abstract

For spacecraft which involve complex attitude or payload control subsystems, and in addition, complex mission operations, a Hardware-in-the Loop (HIL) simulation environment is shown to be an essential and cost effective technique for developing and validating mission software, procedures and operations, and is an essential tool for on-orbit problem diagnosis. This paper presents a case history of the application of HIL simulation to the mission development of the Hughes HS601 family of body-stabilized geosynchronous communications satellites. By integrating the HIL simulation, used in the development, qualification and acceptance testing of the attitude control subsystem, with a real-time, digital simulation of the remaining non-ACS spacecraft subsystems (power, thermal, propulsion, telemetry and command, and payload), a comprehensive, high fidelity spacecraft simulator was developed which incorporated critical flight attitude control electronics hardware and software. The simulator system architecture is reviewed, describing applications of this system to ground station software development and its use for validation and mission operations procedure development and evaluation are discussed. The use of the simulator for real-time mission rehearsals, with the simulator linked to the actual mission ground station is described. Planned use of the system for mission support and on-orbit anomaly investigations is reviewed.

Nomenclature

GSMST	ground station mixed simulation test
MST	mixed simulation test
RTS	real time simulation
HIL	hardware-in-the-loop
ADCS	attitude determination and control subsystem
SCP	spacecraft control processor
CDU	command decoder unit
TEU	telemetry encoder unit
ICST	integrated control subsystem test
HSC	Hughes Space and Communications
ADI	Applied Dynamics International
STMD	selectable telemetry display unit
DAC	digital-to-analog converter
PROM	programmable read-only memory
RAM	random access memory
CLP	communications link processor
DSS	dynamic satellite simulator

Introduction

With the expanding development of powerful, space-qualified microprocessors, a new generation of sophisticated spacecraft attitude and payload control systems has emerged. These systems are providing additional capabilities that were previously prohibitively expensive to incorporate in on-board systems and which were relegated to ground controlled functions. Typical of these new capabilities are,

1) on-board satellite bus management (e.g. autonomous battery charge control, automatic payload reconfiguration, thermal control),

2) sophisticated fault detection and response ,

3) extended autonomous operations,

4) on-orbit performance adaptation and optimization by processor reprogramming.

The primary characteristic of these new capabilities is that they are almost all implemented in software within the on-board microprocessors.

In addition to space processors, the advanced computer workstation has provided enhanced capability for a more sophisticated ground control segment, which can provide a simpler operator interface to the spacecraft and reduce the ground workload. Spacecraft monitoring and operation via automated procedures and expert systems are now practical. Detailed trend analysis and spacecraft performance evaluation can be easily integrated into a ground system.

While the capability exists to build a sophisticated system to match a complex spacecraft bus, the issue of design verification and validation of these heavily software-based space and ground segments becomes crucial. Key to successful development of these systems is the use of high fidelity simulation, including hardware-in-the-Loop (HIL) simulation.

This paper presents a description of the validation system developed for the new Hughes HS601 family of geosynchronous communications satellites. Dubbed the 'Ground Station Mixed Simulation Test' (GSMST) , this system provides an extremely high fidelity real-time simulation environment which integrates actual flight computer hardware and software with detailed computer models of the spacecraft dynamics, space environment, and additional bus and payload subsystems. The GSMST is designed to link directly into either a local computer system, emulating the spacecraft operators ground station, or via phone lines, to the actual ground station itself. The GSMST has become the primary tool used in verification and validation of the space and ground segments and was expanded to provide a sophisticated rehearsal environment for training the mission team.

Section I of the paper presents brief summaries of the HS601 spacecraft , its attitude determination and control subsystem, and the ground segment elements developed for HS601. Section II defines the ADCS development process used for HS601 and describes the role played by real-time simulation in that process. Section III provides a detailed description of the GSMST architecture, design, elements and implementation. The final section summarizes the various applications in which the GSMST system has been used.

I. HS 601 System Description

In 1988 Hughes Space and Communications (HSC) began development of an all new, body-stabilized geosynchronous communications satellite bus, HS601.

ADCS Description

The HS601 attitude determination and control subsystem (ADCS) comprises the sensors, control actuators, and microprocessor hardware and software required to control vehicle attitude during all phases of the mission, including ascent, sun and earth acquisition, normal on-station operations and orbital stationkeeping. The ADCS design is based on three-axis control for on-orbit operations, with spin stabilization during the geosynchronous transfer orbit phase. The ADCS supports antenna deployment, solar wing positioning, autonomous spacecraft management, and failure detection and response functions that allow the spacecraft to maintain service with minimal ground control activity. Major components of the system include redundant spacecraft control processors (SCP), a redundant staring earth sensor, passive, fan-beam sun sensors, dual three-axis inertial reference units for body rate and position sensing during maneuvers, and redundant, two-axis gimbaled momentum wheels. Attitude control in all three axes is maintained by controlling the momentum wheel speed and gimbal angles. A magnetic torquer and solar wing adjustment technique, both controlled by the SCPs reduce the excursions of the wheel gimbal angles and limit thruster firings to those required for orbit corrections, typically every 14 days. Antenna positioning mechanisms provide for optimization of antenna beam pointing. The SCP significantly simplifies command requirements by autonomously performing all transfer orbit, acquisition, and on-station maneuvers through control center initialization of preprogrammed command sequences.

Two key features of the ADCS design are,

1) development of an advanced on-board computer, the spacecraft control processor which utilizes the Marconi processor executing the MIL-STD 1750A instruction set. Programming of the SCP was done in the ADA language in order to develop a standardized, reusable set of flight software,

2) providing the SCP with direct access to both the telemetry and command subsystems, i.e., the SCP can asynchronously request data through the telemetry encoder unit (TEU) from any telemetry point on the spacecraft and can autonomously send commands via the on-board command decoder unit (CDU) to any unit on the spacecraft.

These features provide the basis for the spacecraft management, autonomy, and fault detection provided for HS601.

Ground Segment Elements

To support the new HS601 spacecraft product line a sophisticated ground segment was developed. The two elements of the ground system comprise:

1) a real-time system providing all the tools \necessary for monitoring and controlling the spacecraft, and

2) the Dynamic Satellite Simulator (DSS) which is a tool for operator training.

Real-Time System

The ground real-time system consists of the computer hardware and software necessary to receive and process the down link spacecraft telemetry data and to provide the operator interface for satellite control. The real-time system contains the telemetry decoding and command generation data bases to translate both the incoming telemetry bit stream into engineering data, and to transform engineering information into a binary command uplink. Integrated into the real-time system are automated procedures to execute spacecraft functions (orbit adjustment maneuvers, payload configuration etc.), graphics tools to process and plot real-time and archived telemetry data, display of telemetry data in engineering or graphical formats, and perform long term monitoring and trend analyses. The system also hosts the orbital analysis software used for spacecraft attitude determination throughout the mission, and used to perform all transfer orbit and on-orbit maneuver planning. The real-time system processes tracking station ranging data for orbit determination and generates pointing data for tracking station.antennae.

Dynamic Satellite Simulator

The DSS emulates the complete satellite command path, starting at the satellite control computer, through the command uplink, satellite response, telemetry down link, and back to the satellite control computer frame sync port. The DSS contains functional representations of all HS601 elements including:

- Normal (sunlight) operation
- Eclipse operation
- Large assortment of anomaly conditions
- All subsystem functions

Communications payload
Attitude control subsystem (ACS)
Electrical power subsystem (EPS)
Thermal control subsystem (TCS)
Reaction control subsystem (RCS)
Telemetry and command subsystem

The DSS executes on an HP 9000 series engineering workstation, running a UNIX™ operating system with real-time extensions.

Command Reception

The DSS accepts commands as issued to the command generator from the satellite control computer. The command reception module implements the same handshaking that normally occurs between the units. This module emulates the command decoder unit (CDU) and telemetry items associated with valid and invalid command reception. Uplink configuration commands are entered through the DSS user interface and change telemetered unit states appropriately.

In general, the simulator responds to commands at four levels of detail. The first level of response is in the TC&R subsystem. TC&R status bits (such as accept/reject) are set or reset. The second level of response is the digital status of the subsystem being commanded. Status bits (such as on/off, A/B, enabled/disabled) are set or reset, and SCP registers are loaded. The third level of response is functional verification such as solar wing movement, battery charge initiation, and jet firings. The fourth level of response includes changes in power, sensor, and thermal parameters due to spacecraft configuration and orbit state.

Command Response Processing

A command response exists for each spacecraft command. This command response specifies the telemetry parameters to be modified and the models that require notification of command reception. The command response itself is a custom language designed specifically to support the modeling of spacecraft.

This module, in conjunction with the command reception module, completely supports both delayed execute commands and SCP stored commands. SCP commands can easily be routed to either SCP for immediate or time tagged execution.

Telemetry List Manager

This module maintains the telemetry data base that determine the contents of each telemetry frame. The two telemetry streams may be in any of four predefined configurations, and each stream may be independently configured for dwell mode.

Telemetry Output Processing

The two telemetry streams (normal and dwell) are issued to the real-time computer in the same format as that issued by the frame synchronizer. This module also simulates normal handshaking between the real time computer and the frame synchronizer.

Anomaly Simulation

In addition to simulating normal on-orbit satellite behavior, the DSS provides selected representations of satellite anomalies. Conditions representing contingency and emergency situations may be simulated along with the scenario or scheduled after scenario initiation.

The anomaly control functions permit anomalies to be started immediately or be scheduled to start at any time. Up to ten anomalies may be scheduled at a time. When initiated, each anomaly may 1) change the value of any telemetry variable or internal status value, 2) dynamically reconfigure satellite responses to any command or set of command, or 3) alter the modeling of payload or spacecraft subsystems.

The anomaly page function displays a telemetry page specifically tailored to telemetry relevant to the currently executing anomaly.

II. ADCS Development Process

Key to the software development process is the use of pure simulation and real-time simulation (RTS). Early development of detailed spacecraft dynamic models, sensor and actuator models, emulating the actual hardware elements, with analytic models of the control algorithms provide the basis to support the design. These pure (non real-time) simulations provide for early verification of the ADCS design and performance.

Following development of the SCP breadboard hardware a *mixed simulation* test (MST) phase was entered. The MST environment is the standard technique used to support development of control subsystems for all HSC satellites.

The ADCS mixed simulation test system developed for HS601 comprises hardware and software elements designed to create an extremely high fidelity simulated on-orbit test environment, operating in real-time, for subsystem level development, qualification, and acceptance testing of the spacecraft control processor (SCP). The MST system is capable of operating with either breadboard or flight SCP units and is capable of simulating all mission phases. Key features of the MST system include:

1) Hardware and software to generate all ADCS commands and process all SCP generated telemetry,

2) All interfacing signals for command, telemetry, sensors, actuators, thrusters, and

power designed to emulate the spacecraft interfaces,

3) Access to all signal I/O for SCP and internal software variables and constants,

4) Electrical loads for all outputs of the SCP, designed to match the impedance of the external unit input

5) Spacecraft dynamics, simulated on an Applied Dynamics International (ADI) AD 100 real-time simulation system, which provide data to the test electronics from sensor simulations for the SCP and receive signals from the tester, representing actuators and thruster commands from the SCP for dynamic closed loop operation,

6) Capability to interface all real units that connect to the SCP,

7) Where possible, the MST electrical harness emulates the spacecraft harness in length, impedance, and implementation,

8) Capability for either static (open loop) operation or dynamic (closed loop) operation,

9) Capability to integrate a 3-axis motion simulator for static and dynamic hardware and hardware/software interface verification,

The MST system is designed to support all phases of attitude control subsystem testing.

MST Configuration and Functional Description

A detailed functional block diagram of the HS601 MST system is shown in Figure 5. The MST system comprises:

1) A three-bay rack of special purpose test equipment containing SCP interface panels, telemetry and command interface, sensor simulators, actuator emulators, and other spacecraft interfaces emulators.

2) The AD 100 real time simulation system (with Microvax host computer) and its external interface panels

3) An HP 9000-400 test computer system complete with peripherals for test control and data logging

4) A PROM simulator (controlled by the HP 9000 computer) providing monitoring access to internal SCP microprocessor data

5) Interface hardware and software to communicate between the VAX and HP 9000 for data transfer and data output

6) Support test equipment, including function generators, counters, digital voltmeters, and oscilloscopes

The MST configuration includes ADCS sensor and actuator redundancy and has the capability of operating two SCP units simultaneously. The inputs to the SCP interface panels are controlled through the computer interface panel either statically using the HP 9000-400 computer or with manual input or dynamically using the AD 100 simulation computer.

The AD 100 digital simulation system provides simulated spacecraft dynamics. The computer interface panel in the tester receives scaled data representing outputs from spacecraft sensors modeled in the AD 100 and transfers the data to the electronic interface panels for conversion to the appropriate signal characteristics required by the SCP interface. SCP generated commands are sent to actuator interface panels and then on the AD 100 in analog or digital form. The SCP is thus connected closed loop to the "spacecraft" in the AD 100 simulation computer.

Data from the AD 100 is captured in files for statistical analysis and plotting with a wide variety post processing software available . Data are also displayed real time via strip charts and on graphics screen displays. Telemetry data are captured and written to disk using the HP 9000-400 computer. Telemetry data are displayed real-time on the HP 9000, a CRT on the telemetry panel, and using the STMD displays and DACs to stripcharts. Internal data from the SCP are made directly available via the PROM simulator DACs and are logged on the HP 9000.

Besides logging telemetry data, the HP 9000-400 is also used to send commands (serial and pulse) to the SCP; set up interface panel data for open loop testing, such as pulse phasing and pulse widths, frequency outputs (corresponding to gyro rates and wheel tachometer data), and digital magnitude data (converted to analog voltage for acquisition sun sensors and RF beacon track receiver error outputs); and control the SCP processor test access equipment.

General purpose test equipment (function generators, counters, voltmeters and oscilloscopes) is used to monitor SCP outputs, to simulate some nonvarying (or very slowing varying) spacecraft signals (such as pressure transducer feedback) and as debugging tools.

III. Ground Station Mixed Simulation Test System

Midway through the development program for HS601 it became clear that a method to validate the complete end-to-end space/ground system was needed. The primary areas of concern were

1) validation of the ground segment software, operating procedures and data base against the actual SCP flight software,

2) understanding in detail the *as-built* (not *as specified*) ADCS implementation and operations in a mission-oriented environment,
3) preparation and training of the mission team for an all-new mission environment.

Out of this study developed the concept for the Ground Station Mixed Simulation Test (GSMST) environment, an extremely high fidelity, real-time simulator which could incorporate all elements of the HS601 system and address the areas of concern.

The system integrates the three key elements of the satellite system
1) the ADCS mixed simulation test,
2) the core of the dynamic satellite simulator, and
3) the ground segment hardware and software (telemetry and command processing, attitude determination, mission planning, satellite trend analysis software)

The central element of the GSMST is the previously described MST system which provides the spacecraft dynamic simulation along with the SCP and all its interfaces.

The GSMST system expands the basic MST to incorporate the DSS to simulate the remaining portions of the spacecraft bus (electrical power; thermal control; propulsion; and telemetry, command, and ranging subsystems) and payload; and provides the capability to interface the complete simulator to a ground station emulator over a standard digital interface. Features of the simulator include:

1) All external inputs to the SCP are made through the standard connector interfaces; thus, all sensor and command inputs are modeled exactly as on the spacecraft, with actual voltage level, wave shape, and data format simulated.

2) All SCP outputs (telemetry, thruster commands, and wheel and stepper motor drives) are monitored through the standard connector interfaces. Individual stepper motor windings are monitored to verify proper phasing.

3) All critical SCP hardware input/output functions and hardware/software interfaces are included in the simulator to provide a comprehensive end-to-end test capability.

4) The high fidelity dynamics, orbital and environmental models which are executed within the AD100 are identical to the simulation used during the ADCS development and qualification process and provides for precision control of the simulation environment

5) The spacecraft simulation includes all rigid and flexible spacecraft dynamics, articulated elements (antennas, solar wings, and momentum wheel platforms); transfer and operational orbit dynamics and kinematics including sun, earth, and moon geometries; and command outage effects.

Key features of the DSS component of the ADCS simulator include:

1) Routing of appropriate commands to the MST hardware as well as simulation of all non-SCP commands via subsystem command responses and/or dynamic subsystem models.

2) High fidelity models of the electrical power, propulsion, and thermal control subsystems, which are fully integrated with the MST dynamics and SCP hardware.

The elements of the simulator comprise:

1) An HP9000/400 UNIX-based engineering workstation which hosts all simulator software as well as provides the user interface for operation and control of the simulator.

2) The AD100 real-time simulation system which contains all interface input/output) hardware for connection to a flight or breadboard SCP, as well as the *compute engine* which executes the spacecraft, sensor, and actuator dynamic models.

3) The MST interface equipment and harness which provides test access for the simulator and connections between the SCP, AD100, and HP9000 computers

4) Digital Equipment Vaxstations which host the ground station real-time software

5) Commercial equipment (stripchart recorder, laser printer, user terminals) to support the simulator environment, and communications to customer ground stations

The HP workstation provides the platform for all software development as well as a window environment for operating the simulator. During simulator operation the DSS process, running on the workstation is used to send commands to the integrated spacecraft simulator (manually or from files) generate and process two integrated telemetry streams (normal and dwell), log telemetry to disk, generate telemetry displays, and provide a telemetry and command interface to a ground system to support network testing and mission rehearsals.

A laser printer is attached to the computer system to print hard copies of programs, telemetry, command files, command histories, and plots of completed tests. There are programs that plot time histories of several variables simultaneously. The tester system computer displays selected columnar data of unprocessed telemetry variables on a CRT and stores telemetry in a file for post processing. Selected page displays of telemetry in engineering units is available.

Communication of critical vehicle dynamic, kinematics and orbital data between the AD100 and DSS for integration and coordination of all simulation elements is carried out over a dedicated ethernet interface using the Communication Link Processor (CLP) function of the AD100.

The DSS also provides the telemetry and command interface between the GSMST satellite simulator and the ground station. The DSS formats the spacecraft telemetry frame, integrating data from its own subsystem models (power, thermal, propulsion, payload) with SCP data transferred from the MST by communication protocols within the HP9000. Using the RS232 ports on the HP9000 the DSS transmits the formatted telemetry data to the ground station computer at the minor telemetry frame rate. In addition the DSS receives commands sent to the spacecraft by the ground station computer through an RS232 port. The DSS decodes the command data, internally processing those commands that are directed to the vehicle subsystems in its model and transferring ADCS commands to the MST system through internal communications protocols.

GSMST Simulation Models

The total spacecraft model contained in the GSMST is composed of tester emulations of the hardware interfaces to the breadboard SCP, functional dynamic models of the spacecraft, sensors and actuators on the AD100, and non-ADCS subsystems within the HP9000 DSS.

Actuator Models

The AD100 interfaces with the SCP to provide simulation inputs for thrusters, stepper motor winding drives, and momentum wheel torque. Detailed thruster modeling includes turnon and turnoff delays and variable thrust amplitudes. The magnetic torquer is included in the detailed model.

Disturbance Torque Model

Stepper motor outputs are input to the proper actuator models, which integrate disturbance torques into the spacecraft model. The solar wing drives are modeled this way and drive the spacecraft dynamics model, which includes all the rigid and flexible dynamic modes of the solar panels and other bodies attached to the spacecraft. The windmilling and overturning solar pressure disturbance torques are modeled for the solar wings. The sun angles of incidence are calculated from the orbit ephemeris model. Magnetic field interaction, thermal deformation, and thruster plume impingement also form a part of the disturbance torque model.

Spacecraft Dynamics

The spacecraft 13 degree-of-freedom dynamics model is designed for all the spacecraft control phases, including initial ejection and ascent from launch vehicle, spinup for transfer orbit burns, spindown, sun-earth acquisition, stationkeeping maneuvers, and earth pointing. Figure 8 shows the components included in the model. Fuel use is accounted for in the mass property changes, slosh modes, and dedamping time constants. A complete flexible model of the spacecraft is represented by all the modes within the control loop bandpass, including appendages and solar arrays. The mass properties also reflect several spans in the life of the spacecraft, from beginning of life (BOL) to end of life (EOL) with empty fuel tanks. Special ascent/transfer orbit cases are included for worst case fuel slosh.

Orbit Model

An orbit model is incorporated into the simulation using an earth centered inertial (ECI) coordinate system and synchronized to the annual calendar. The orbit model is initialized with an input start day, time, orbit velocity vector, orbit position vector, and spacecraft attitude vector. The sun and moon ephemeris data are a part of the model and dependent on time. Earth/moon eclipses and apogee/perigee crossings can be computed with the model. An alternative specification of the orbit are provided using classical keplerian elements at the epoch.

Sensor Models

Sensor models are high fidelity models incorporating an emulation of the hardware interface to the SCP in the tester and a software functional model in the spacecraft simulation. The spacecraft simulation communicates sensor input to the test interface modules, which translates them into earth sensor, inertial reference unit (IRU), momentum wheel speed, and sun sensor electrical outputs for the SCP. The inputs from the simulation to the tester are made every real-time simulation frame time (4ms). Inputs to the SCP from the tester follow the same timing as the spacecraft hardware. Sun sensors are modeled as a spinning sensor pulse for transfer orbit or analog outputs for acquisition. Earth sensors have spinning horizon crossing indicator (HCI) outputs and static sensor individual cell outputs transmitted over a serial digital interface. IRUs are modeled with pulse and frequency outputs, depending on the mode.

Electrical Power Subsystem (EPS) Model

This model simulates the state of the EPS. It takes into account the effect of the solar panel angles, eclipse condition, battery charge and discharge controller logics, the state of bus voltage limiters, and the spacecraft power load as determined by the on/off state of each unit.

Propulsion Subsystem Model

The propulsion model simulates the flow of high pressure gas in the propulsion system and keeps track of

fuel and oxidizer consumed by the thrusters. It considers the level of propellants in the tanks, the state of latch and squib valves, tank temperatures (determined by the thermal model), and tank volumes to determine the transducer output values.

Thermal Model

This finite element model uses the same data used by Systems Engineering to characterize the real vehicle's thermal performance. This model is rehosted on the HP 9000 computer, configured for real-time use, and interfaced to the telemetry system. It simulates the effect of sun angle on solar panels and units, the effects of unit status, heater output, and thruster activation.

TC&R Subsystem Model

The DSS accepts clear and encrypted commands as issued by the ground control computer. This subsystem models the command decoder unit (CDU) and telemetry items associated with valid and invalid command reception. Uplink configuration commands may be entered through the DSS user interface and will change telemetered units states appropriately.

The telemetry model supports all predefined telemetry stream configurations and rates as well as the dynamic modes inherent in the SCP. When connected to the MST configuration, this model inserts into the telemetry stream information obtained from the SCP. Depending on the SCP mode selected, this information may reflect stored and verify buffer contents, memory readout information, or selected dwell mode parameters.

IV GSMST Applications on HS601

As mission procedures and operational instructions were developed, these were 'practiced' individually using the GSMST. In preparation for the launch the detailed mission sequence of events was developed and exercised using GSMST. During this phase the sequential transitions between control modes was exercised using the actual ground environment.

The GSMST is the center of all mission rehearsals, initially internal rehearsals using the Hughes Mission Control Center in El Segundo and later from Australia, linking the customer ground station in Belrose (which would control the actual mission) to the HSC Control and Dynamics Simulation Laboratory in El Segundo via phone lines. During these final mission rehearsals the customer mission team is included and trained using the GSMST. All mission procedures were exercised in GSMST prior to execution on the actual spacecraft. During the actual OPTUS B-1 mission the GSMST system was kept operational on standby to validate any real-time changes made to the mission procedures prior to execution on the actual spacecraft. The mission control center in Australia could be quickly linked to the GSMST and the modifications tested and verified.

The high fidelity simulation capabilities of the GSMST supported real-time rehearsals which involved verifying all geosynchronous transfer orbit activities, including ground attitude determination, maneuver planning and execution, and mission timeline validation.

Once the satellite was on-station the GSMST was used to verify all on-orbit test procedures prior to running the tests on the spacecraft. The GSMST environment provided the primary tool in development, testing and validation of all on-orbit reprogramming of the SCP including verifying the ground segment software used to transmit and verify the RAM 'patch' to the spacecraft

An additional use of GSMST was to provide a safe, mission oriented environment in which system 'stress testing' could be carried out. This test phase was aimed at 'breaking the code' i.e., to push the 'as built' system to its limits to understand all operational capabilities as well as defining any operating constraints for the system. The high fidelity GSMST provided the ideal environment in which to conduct these 'what if' scenarios.

Additional programs utilizing GSMST are Galaxy, MSAT, ASTRA,and DBS. The significant demand for GSMST has led to development of a HSC mission support system. The HS601 mission support system provides a GSMST facility dedicated to all aspects of HS601 mission preparation and operations. The system is capable of operating locally or linking to a customer ground station anywhere in the world.

V. Concluding Remarks

The use of hardware-in-the-loop simulation has been shown to be an essential tool in design, development and validation of the sophisticated attitude determination and control subsystem for advanced satellite systems. Extending the standard control subsystem development test environment to include the entire spacecraft and ground segments was a natural addition to the existing subsystem test capability. By providing an extremely high fidelity simulation environment with imbedded critical flight hardware and software, the GSMST has proven itself to be an essential and cost effective tool for supporting mission operations.

References

Slafer, Loren, 1988, *AIAA Simulation Technologies Conference*, USE OF HARDWARE-IN-THE LOOP SIMULATION FOR SPACECRAFT MISSION PREPARATION, PLANNING AND SUPPORT

MISSION EQUIPMENT SIMULATION and EVALUATION FACILITY

Louis Roumeliotis, Eng
CAE Electronics Ltd
Saint Laurent, Quebec, Canada

ABSTRACT

In today's world of budget restraints, simulation and rapid prototyping tools are becoming an increasingly valuable means for Military R&D agencies to research and evaluate their options prior to equipment procurement. With this notion, CAE has conceived the Mission Equipment Simulation and Evaluation Facility (MESEF). This facility provides an enhanced simulation environment for R&D agencies to perform research in existing and new aircraft mission equipment packages. In pursuit of new simulation technologies, the MESEF Simulator was procured by the Crew Station Research and Development Facility (CSRDF) for simulation development and research.

This paper gives details of the MESEF design requirements, and the software and hardware tools that were developed to satisfy these requirements. It is concluded that such devices as the MESEF cockpit provide a means for rapid equipment evaluation prior to in-service testing.

INTRODUCTION

The MESEF simulator is a new type of research platform that permits the user to reconfigure the airframe and sensor suite, as well as the controls and displays via a series of software and configuration management tools. These tools enable the user to create a library of aircraft configurations, each defined with a unique set of capabilities and characteristics.

A software development package (ROSE) provides the researcher with a graphical user interface environment to model individual components of the simulated aircraft. Once the individual components have been created, they can be linked together to create a complete aircraft simulation. These tools can be used to develop a full complement of high fidelity generic aircraft systems including navigation, autopilot, flight, avionics, tactical and weapon systems.

Any aircraft in the library can be selected and operated within a comprehensive tactical environment simulation. The current system uses CAE's Interactive Tactical Environment and Management System (ITEMS). ITEMS provides the researcher with the capability to define tactical scenarios with varying complexity in which the aircraft can be exercised and/or operated.

This paper describes the components comprising the MESEF simulator including the generic cockpit layout, the distributed computer architecture and the software development environment. A protype aircraft simulation was configured using the MESEF tools as a proof of concept. This protoype configuration will be used to demonstrate the capabilities and the potential of a simulator device such as MESEF.

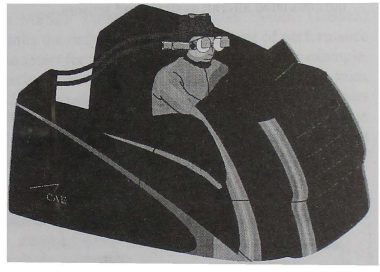


Figure 1: MESEF Cockpit

RECONFIGURABLE COCKPIT

Generic Cockpit Cubicle

One of the main components of the MESEF complex is the generic cockpit. The cockpit provides the pilot with control of the aircraft systems via a series of controls and displays.

The cockpit consists of the following:

- Single aircraft seat with up/down and fore/aft adjustments.
- Displays:

Copyright © 1997 by the American Institute of Aeronautics and Astronautics, Inc. All rights reserved.

- Front Touch Sensitive 26" color screen to display and control the main cockpit systems and instruments.
- Side Touch Sensitive displays installed on each side of the cockpit for display and control of aircraft sub-systems.

- Four (4) degree of freedom hand controllers installed on each side of the cockpit. The configuration of the handcontrollers can be setup to provide system/sensor controls and/or flight controls (including collective and cyclic capability).
- Fibre Optics Helmet Mounted Display (FOHMD) visual system including a head and eye tracking system. Superimposed on the visual image is a user defined Head-Up-Display (HUD). The HUD includes flight, navigation, sensor and weapon symbology.

Computer System Architecture

The computer complex comprises of a network of computers each dedicated to a specific function:

- Main computer: Hosts and executes the software of the simulated aircraft systems and sensors.
- Software Development Workstation: Provides the graphical software development environment for the creation of aircraft simulations. The software can be developed, generated, executed and tested in standalone mode prior to being downloaded to the other computers. The MESEF DBMS also resides on this computer.
- Cockpit Displays Computer: Processes the cockpit panels displayed on the 26" touch screen display in the cockpit.
- ITEMS / EOC Computer: Hosts and executes the tactical environment. The console serves as an Experimenter/Operator Station. This station provides the operator with controls and displays to monitor and evaluate the performance of the pilot as the scenario executes in real-time.

Real-time communication is used to exchange simulation data between the different computer platforms. The communication technology used includes both shared memory communication (SCRAMNet) and network communication (ethernet). Figure 2 depicts the lay-out of the MESEF simulator platform.

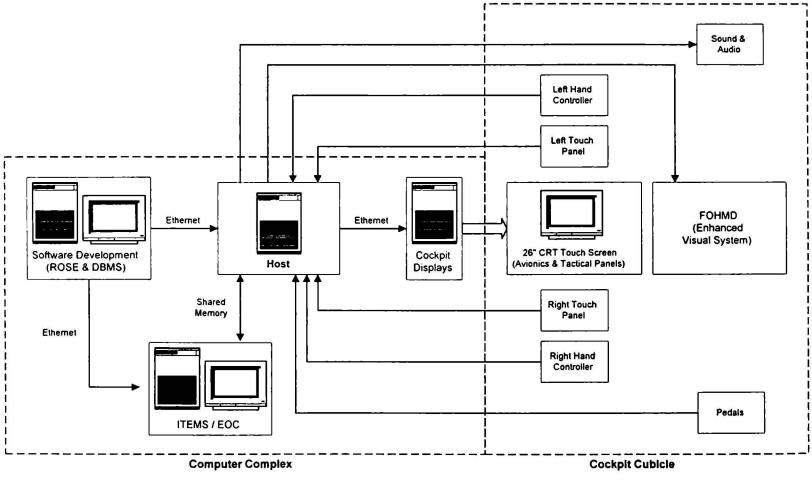


Figure 2: MESEF Complex

Visual System

CAE's Fibre-Optic Helmet Mounted Display (FOHMD) provides the pilot with a high resolution, full color and collimated computer generated visual image. The two channel image generator (one per eye) provides a high resolution image of 2400 x 1800 pixels (4Mpixels) per eye. To increase realism, full color texture is added to the scene as well as micro-texture for added low-altitude cues.

Each IG channel generates a dual-acuity digitally merged image consisting of a high resolution inset and a wide field of view background. The instantaneous direction of the FOV is aligned with the orientation of the pilot's head as measured by the head tracking system. The position of the high resolution inset is in the direction of the pilot's gaze as measured by the eye tracking system. The resulting two-eye FOV is 120 degrees horizontally by 55 degrees vertically with a field of regard of 360 degrees.

The FOHMD provides the pilot with a full out-of-cockpit view and a superimposed "cockpit mask" to represent the physical cockpit structure. The "cockpit mask" is a silhouette of the cockpit structure and it is always perfectly aligned with the cockpit. The imagery overlaid on this silhouette is video masked to permit the pilot to view the interior of the cockpit (including the aircraft instrument displays) through the helmet optics. Overlaid on to the visual image is the Head-Up-Display (HUD) symbology to provide relevant flight, navigation and tactical information to the pilot.

The FOHMD supports the effects of environmental conditions and the display of a variety of moving models including ground, sea and air targets, missiles, rockets, ground explosions, air explosions, burning ground targets and tracer fire.

To simulate thermal imaging systems, the FOHMD has an advanced thermal environment model which can produce and display Infra-Red (IR) Imagery. The functions that are supported include polarity (black hot or white hot) , FOV (wide, medium, narrow, zoom), brightness, contrast, auto gain and level control. Overlaid onto the image is relevant IR system symbology.

SOFTWARE DEVELOPMENT ENVIRONMENT

CAE Electronics' Real-time Object-oriented Software Environment (ROSE) combines powerful development, debug and system integration tools into an integrated visual software programming environment. The graphical user interface (GUI) is X-Window/Motif compliant and can be used on a graphical workstation, or X-terminal. The suite of software tools provide a graphical, icon-based, object-oriented software development environment. The client/server database allows many developers to work simultaneously on the same system. ROSE also provides an integrated runtime environment which facilitates integration and testing.

The ROSE editors allow the user to model a system by placing objects in a schematic, where each object has an associated fully dynamic graphic display (icon). Hence, to model a system, the user selects icons (representing standard components) and then links them to construct a graphical representation of the system to be simulated. For control systems, for instance, objects such as integrators, summers and filters are logically linked together to create schematics. The resulting schematics have the appearance of actual control system simulation block diagrams.

The MESEF Database Management System (DBMS) permits the definition of a complete set of performance parameters for a particular type of system. For example, upon integrating a radar model to the aircraft sensor suite, the user also assigns to it a set of performance parameters such as transmission power and antenna gain. The same radar model can be integrated into a number of different aircraft configurations with each given a different set of parameters.

ROSE consists of the following main components:

- Editing tools:
- Code Generators
- Multi-User Repository
- Workstation test environment
- Site integration and runtime environment

ROSE Editing Tools:

The ROSE Editing tools allow users to create libraries of objects, and to construct schematics using these objects (refer to Figure 3).

Graphics Editor:

The Graphics Editor is used to design graphic icons and virtual panels with associated input and output dynamic information. It is used to create numeric and text readouts, and realistic meters, gages, buttons and switches.

Object Editor:

The Object Editor is used to develop an object library. The objects are the lowest level building blocks for the design of simulation models. A ROSE object consists of a graphical icon, variable declarations and pseudo code.

Model Editor:

The Model Editor is used to create simulation models using schematics. This is achieved by dragging objects from the library windows onto the schematic window. The objects are then connected to one another.

Builder:

The Builder feature is used to place schematics into groups and to generate code. The groups are placed in a configuration for realtime execution. A simulation configuration (including the sequence of exexution of the groups) is set up graphically.

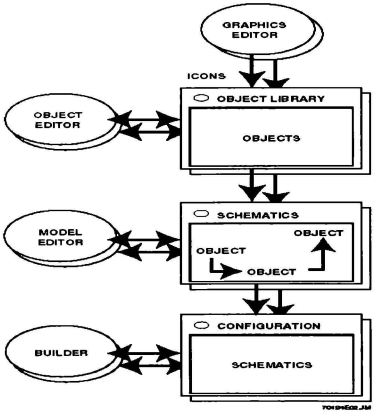


Figure 3: ROSE Editing Tools

ROSE Code Generator

The Code Generator is used to generate the code of the simulation model in C, Fortran or Ada.

ROSE Multi-User Repository

The multi-user repository provides a central location for storing the data of the different configurations and making it available to any designer using the ROSE tools. The repository can be installed on either a standalone workstation or on a server connected to a multi-user, multi-workstation environment via LAN.

ROSE Workstation Test Environment

The Runtime Executive is used to load and run a configuration. It consists of a suite of stimulation and monitoring tools which allow a user to graphically open schematics, and to display, plot and modify any value of any simulation variable.

Site Integration and Runtine Environment

The site integration and runtime environment is used to integrate complex simulations on the test site. It provides configuration control and it is the mechanism which ROSE uses to transfer schematics to the test site.

Figure 4 depicts the ROSE development environmnet.

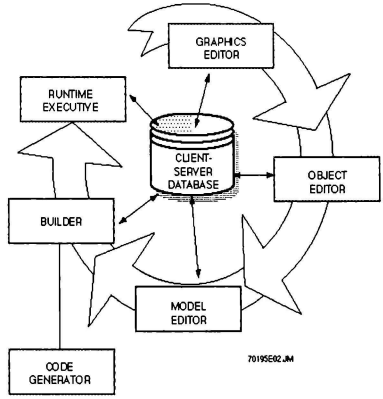


Figure 4: ROSE Environment

USING ROSE ON MESEF

Aircraft System Simulation

A prototype aircraft simulation was developed by CAE to demonstate the potential of the ROSE software development environment. Many of the aircraft systems were modelled using the ROSE tools whereas existing systems were interfaced to the systems developed by

ROSE; i.e. ROSE permits legacy software systems to be integrated to systems developed by ROSE. The prototype aircraft included the following systems: Flight Model, Flight Control System, Propulsion System, Navigation System, Tactical Systems, Weapons, Radar, etc. Figure 5 and figure 6 provide a top-level view of the avionic and tactical systems respectively.

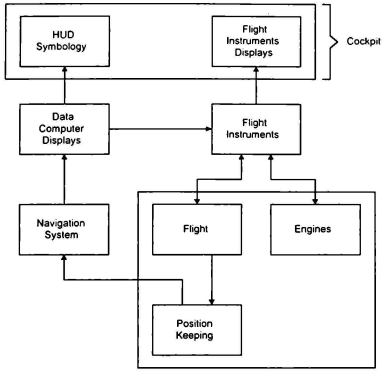


Figure 5: Avionics Block Diagram

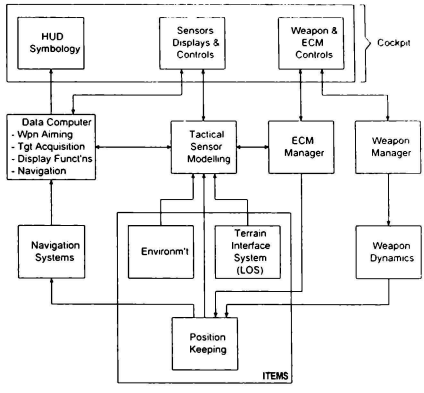


Figure 6: Tactical Systems Block Diagram

Head-Up-Display (HUD)

The Head-Up-Display is the primary display for flight control information. The information is displayed to the pilot in the FOHMD as an overlay on the high resolution inset. ROSE provides the software tools to layout and to configure the symbology on the HUD as desired. This includes navigation, flight, tactical and weapon symbology. The HUD symbology is integrated to and driven by the aircraft simulation systems.

Cockpit Displays and Controls

The cockpit displays and controls are presented to the pilot on a 26" touch screen monitor and two side touch panel displays. For the prototype aircraft simulation, the display area on the monitor was subdivided into four identical windows and a Warnings Display area. For each window there is a main menu area (selections at the top of each window), which allows the pilot to call up any one of the main system display panels. The Panels that can be selected for display in the cockpit include Weapon, Target Acquisition, IR Tracker/Laser, FLIR Controls, Radar, RWR, Jammer, Chaff/Flare Dispensers, Tactical Map, Flight Instruments, Engine Indicators, IFF and Communications. Figure 7 depicts a typical setup selected by the pilot on the 26" monitor in the cockpit.

The software programmable Electro-Luminescent (EL) side touch panels provide addional controls for the aircraft subsystems. Figure 8 depicts a typical configuration of the EL panels.

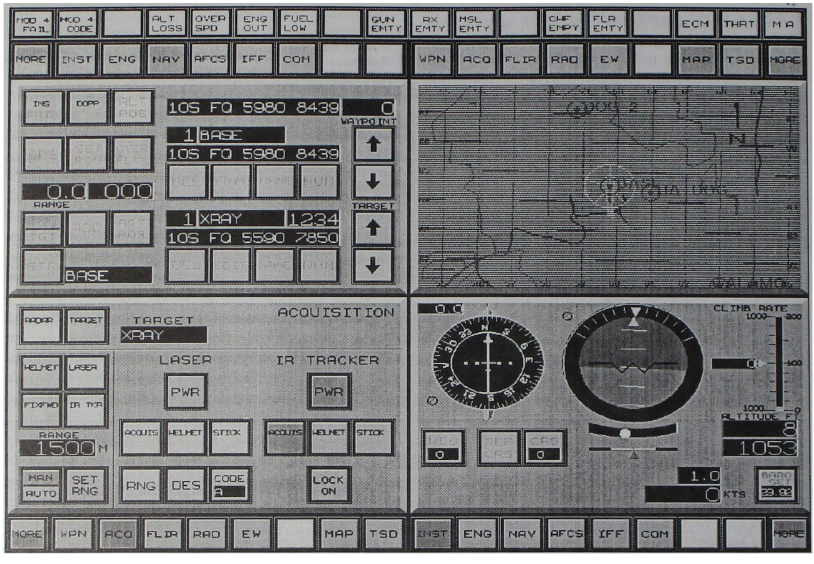


Figure 7: Cockpit Panel Displays

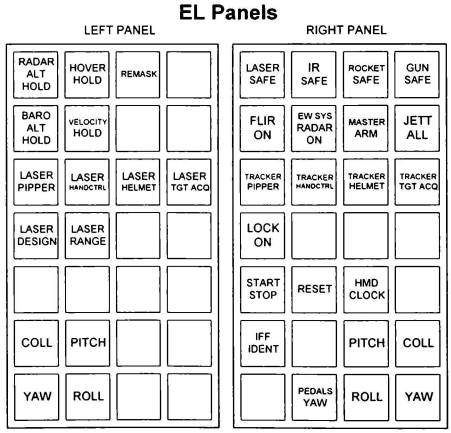


Figure 8: Side Touch Panels

TACTICAL ENVIRONMENT

A representative "real world" tactical environment is an essential feature on any simulator device such as MESEF. It provides the capability to operate and exercise the MESEF and its sensor suite in a tactical environment setting. CAE's Interactive Tactical Environment and Management System (ITEMS) provides an enhanced tactical environment simulation where both friendly and opposing forces can interact with each other and with the simulator device and its sensor suite.

ITEMS provides a user-friendly graphical interface to allow the user to create and store scenarios, vehicles and vehicle doctrine in the ITEMS DBMS Databases. Once a scenario has been defined and stored in the databases, it can be selected and executed in real-time. The MESEF cockpit can be "coupled" to ITEMS allowing the pilot to carry-out full mission exercises.

ITEMS is sub-divided into two main functional components:

- ITEMS Database Management System (DBMS): A background task used to define tactical scenarios.
- Scenario Execution Function: A foreground task used to execute a tactical scenario in real-time.

ITEMS Database Management System

ITEMS DBMS provides the user with specialized editors to define complete mission scenarios. The GUI uses a menu-driven window-based format to provide a user-friendly data entry environment. In addition to the easy-to-use speciliazed editors, visual tools such as a 2D map of the gaming area and a 3D display enable the user to easily layout the missions and objectives, and to piece together scenario elements as required for the training exercise or experiment. A complete scenario contains all the information to execute a complete tactical exercise in real-time. Tactical scenarios are created off-line at the ROSE/DBMS workstation and are downloaded to the ITEMS computer for real-time execution.

The ITEMS DBMS databases are ordered in a hierarchy. The elementary elements defined in the lower level databases are used to create higher level elements. For instance, a vehicle definition is comprised of its physical and dynamic characteristics and its sensor suite. Its sensor suite is assigned by referencing elements (such as radar, weapons) defined in the lower level databases. The highest level database is the scenario database. The scenario database contains the complete definition of a scenario including the physical area of operation (gaming area), atmospheric conditions, vehicles (selected from the vehicle library), and vehicle behavior. Figure 9 depicts the ITEMS database hierarchy.

Vehicles in a scenario are physical entities of tactical interest detectable by the simulated aircraft systems/sensors. The simulated aircraft equipment will detect vehicles only if within the acquisition envelope of its sensor suite. Vehicle types supported by ITEMS include fixed wing and rotary wing aircraft, surface and sub-surface vessels, and ground platforms. ITEMS models each vehicle in terms of its physical and dynamic characteristics, its systems/sensor suite and its behavior in the scenario. The vehicle system/sensor suite can include

- radars
- warning receivers (radar, laser, IR)
- lasers (designators, rangers)
- sonars
- sonobuoys
- communications (radios, datalink, direct link, underwater telephone)
- jammers (radar, IR, acoustic jammers)
- countermeasures (chaff, flares, acoustic decoys)
- weapons (missiles, guns, rockets bombs, depth charges, torpedoes)

The interactive behavior of a vehicle is modelled using a knowledge-based system. This knowledge-based system uses rules to control the vehicle's actions and reactions within the evolving tactical scenario. Rules define the vehicle's mission, how and when it will use its systems, and the vehicle's response to a given threat situation (i.e. fire weapon, perform evasive maneuver, dispense chaff ...). The rules continuously evaluate the conditions of the threat environment based on information attained by the vehicle's systems/sensors to determine the vehicle's response to the current tactical situation. As circumstances change so will the actions and reactions of the vehicle. As a result, the rule-based system provides a fully interactive environment, where each vehicle responds automatically to other vehicles in the gaming area and to the piloted MESEF aircraft simulation.

ITEMS Scenario Execution Function

Any tactical scenario created and stored in the DBMS databases can be invoked for a training session or experiment. Upon invoking a scenario, the Scenario Execution Function initializes the scenario and then proceeds to execute it automatically in real-time according to the definition of the scenario.

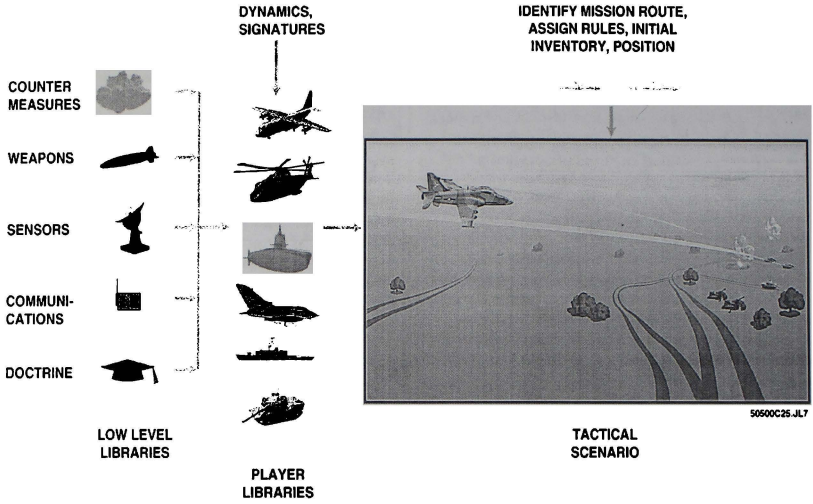


Figure 9: ITEMS Database Hierarchy

The DIS networking capability of ITEMS permits the MESEF simulator to be networked to other simulator platforms in the same facility or to other simulator platforms in remote geographical locations. This capability enables the MESEF simulator to participate in joint tactical exercises. ITEMS is currently being upgraded to support High Level Architecture (HLA).

RUNNING AN EXPERIMENT

Prior to loading the run-time simulation, the user must first select the aircraft configuration from the configurations library. Once selected, the appropriate runtime software configuration is loaded for execution. When loaded, the aircraft cockpit assumes the characteristics and capabilities as they were modelled using ROSE.

Once the simulation is loaded, the Experimenter/Operator Station (EOS) provides the user with runtime controls to initiate an exercise. The following selections are required:

- An Aircraft Initial Setup from a list of setups defined in the MESEF Aircraft Setup Database which provides the initial setup of the aircraft such as initial fuel weight and weapon load.
- A tactical scenario from the list of scenarios defined in the ITEMS Scenario Database.
- Activate the 'RUN' button.

At this stage, the aircraft systems and sensors are executed in real-time. The systems interact with the cockpit displays and controls, and drive the HUD symbology. The aircraft simulation also interacts with the evolving tactcial environment.

The EOS provides the user with displays and controls to monitor and evaluate the performance of the piloted aircraft in the evolving tactical scenario. Additional controls permit the operator to manually intervene and thus to alter the normal evolution of the scenario.

DATA RECORDING AND ANALYSIS

The Data Recording and Analysis (DRA) facility permits the user to record scenario data and events as they occur during scenario execution for later analysis and study.

DRA supports data reduction through the use of rules. The rules, entered via a specialized DRA Rule-Base Editor, define the recording control information. Each rule is comprised of a condition block which defines the scenario event(s) to initiate a recording and a 'recording list' to specify the simulation data to be recorded. The 'recording list' can consist of any simulation variable(s) which can be recorded in one of three mode: single shot, triggered or continuous.

During scenario execution, the action-oriented rules are continuously evaluated and the data is recorded as specified by the rules. On completion of the exercise, the recorded data stored on disc (or magnetic tape) can be transferred to a relational database management system for display and analysis.

CONCLUSION

This paper is best ended by a statement in a post evaluation report: "The MESEF cockpit which was ostensibly conceived and procured as a research tool, has evolved into a very useful and impressive operational mission simulator. With the Visual System, Fibre-Optics Helmet Mounted Display (FOHMD) and FLIR, this is a very formidable device to researchers and aircrew alike."

REFERENCES

A) ROSE: A Visual Programming Approach to System Design
Peter Ahn, Nick Moscato, Trevor Dudley, Richard Elton
CAE Electronics Ltd, PO Box 1800
St Laurent, Quebec, Canada, H4L 4X4
e-mail: rose@cae.ca

ACKNOWLEDGEMENTS

The author wishes to acknowledge the effort of the engineers of CAE Electronics who worked so hard to make the MESEF project a success, notably Peter Mazzei, Pierre Cardinal, Guy Beauchesne and Dave Rowe.

AIAA-97-3809

SIMONA - A RECONFIGURABLE AND VERSATILE RESEARCH FACILITY

Sunjoo K. Advani*
Etienne van Zuylen†
Sander Bettendorf‡

International Centre for Research in Simulation, Motion and Navigation Technologies,
Delft University of Technology, Anthony Fokkerweg 1, 2629 HS, Delft, The Netherlands, s.advani@lr.tudelft.nl

Abstract

This paper reviews the development of the International Centre for Research in Simulation, Motion and Navigation Technologies, SIMONA, a program of the Delft University of Technology, based in The Netherlands. A primary goal of SIMONA is to investigate the interaction between humans and flight vehicles in a synthetic environment, and thereby address pertinent problems in the development and application of simulation technologies. Central to SIMONA is a high-bandwidth, reconfigurable research simulator. This device (the SIMONA Research Simulator, or SRS) provides to its occupants very high-fidelity motion cues through a six-degrees-of-freedom hydraulic motion system and a wide-angle continuous collimating visual display system. The flight-deck is configured inside a composite-materials shell which serves also as the primary structure of the moving platform. The paper describes the purpose and the status of SIMONA, outlining the recent goals achieved in the process of realizing this unique facility. In particular, the initial results of the motion system calibration tests are described. The paper concludes with a projection of expected development milestones and research programs.

* Director of SIMONA, Assistant Professor, Member AIAA

† M.Sc. student, Fac. of Aerospace Engineering

‡ M.Sc. student, Fac. of Mechanical Eng. and Marine Technology

Copyright © 1997 by the American Institute of Aeronautics and Astronautics, Inc. All rights reserved.

Introduction

The skills which a vehicle operator acquires and reinforces during the training process directly influence the individual's behaviour during operational tasks. Training is increasingly shifting towards simulated environments which, in turn, are systematically *replacing* in-flight training. In commercial aviation, current simulation techniques permit zero-flight-time recurrent or type-training, a process by which pilots with qualification on a particular transport aircraft type can transition to another transport aircraft type, without the need to exercise any training flights. The modern simulator provides a suitable environment for this type of activity. In the near future, aviation operators intend to proceed further, by transitioning candidates with experience in basic piston-engine training aircraft, through a simulator flight training program, and to the transport aircraft, eliminating the need for training in the transport aircraft. This process is called *ab-initio zero-flight-time training*.

Through training activities like that mentioned above, new questions will be imposed on the airlines operating and benefiting from these training devices, on the certification authorities who are responsible for upholding the safety of transport vehicles, and on the simulator manufacturers who produce these systems. For example, when a particular training application is required, how should the resulting simulator requirements be specified? What training value does a particular training device have, as opposed to training in the flight vehicle? Helping solve these problems is the main objective of SIMONA.

Manual control of flight vehicles

Despite automation in the cockpit, accurate closed-loop *manual* control capability of the vehicle by the pilot remains essential. It is in fact the *control of the vehicle* by the pilot which we try to synthesize and match between the simulator and airplane. Eventually, the pilot in the simulator should be able to control the simulated vehicle in much the same way as he would in the aircraft.

The flight of a vehicle involves its controlled movement through space. This control requires feedback from a system that regulates the vehicle state by comparing its actual motions with respect to the desired goal, and applying corrective control. Control by the human pilot requires the sensation of the vehicle motions through the his sensory systems, followed by decision-making, response selection, and response execution processes. Note that it is the responsibility of the simulator designer to provide correct analogies of the vehicle sensations in the simulator. The training program must adequately fill in the remaining elements.

The perception of motion, required for manual control, involves the acquisition and processing of information from a variety of sources: The vestibular system of the inner ear senses specific forces and angular accelerations, while the visual system detects motion of the environment relative to the observer. The vestibular and the visual systems which dominate in the perception of self motion. The perception of motion through the peripheral visual field is considerably quicker, but less accurate, than that through the central visual system, due to the shorter neural pathways to the Central Nervous System of the latter[1].

Cueing in flight simulators

The simulator is a system of devices which synthesizes the aircraft stimuli through mathematical approximations, and through electronic and mechanical media which attempt to re-introduce the sensory information available in the aircraft, to the various perception channels of the pilot[2]. The primary environmental motion stimuli are synthesized by a visual display system, which computer-generates an image of the world outside the cockpit, and projects this picture onto a screen which the pilot views. Modern visual displays provide optical cues which are nearly identical with respect to those seen from the aircraft cockpit. The scenery is highly-convincing in resolution and in field-of-view. In fact, large field-of-view displays on their own can induce a perception of motion, particularly at lower frequencies.

As a consequence of its mechanical limitations, the simulator *motion system* cannot however physically reproduce the aircraft inertial translations and orientations with equal gain. The platform payload mass properties can degrade the system's phase margins. *Rational compromises need to be made.* The generation of adequate inertial motion cues in the simulator remains a challenge. The fact that it is possible to give a pilot the feeling of flying his own aircraft within the limited space of the simulator is due to the limited ability of humans to detect differences between environmental (visual) and inertial (whole-body) motions, up to some extent[3]. In our world, the perceived inertial motions always move in conjunction with environmental motions.

Simulator fidelity

One could propose four levels of simulator fidelity, as follows:

Fidelity Level 1

The *ideal* situation would be when the simulator motion stimuli are perceived exactly as they would be perceived in the real aircraft, requiring the forces acting on the human pilot be identical in both magnitude and phase to those in the aircraft, throughout the duration of the simulated flight. Even for short periods of time, however, this is not feasible.

Fidelity Level 2

Instead of providing *perfect* reproduction of the *stimulus* signals present in the aircraft, one could present the motions of the vehicle to the simulator pilot such that there is *no adaptation* required by the pilot when the control behaviour learned in the simulator are applied in the aircraft, or when the simulator is "flown" by the same pilot in the aircraft. The differences between the aircraft-generated and simulator-generated stimuli then would not influence a change in pilot control behaviour.

Fidelity Level 3

If the above is not achievable, one could propose an environment in which there *is* adaptation, but to such an extent that it does not have *negative* effects on the controller. The adaptation required would be corrected either during the learning process, or during the eventual application of the skills in the vehicle. In this case, the skills learned should have only a positive influence on the controller's perception of the aircraft dynamics. While there would be a difference between the pilot control behaviour in aircraft and the simulator, the training goal could still be achieved.

Fidelity Level 4

The situation can however be worse, if the stimuli presented cause *incorrect* control behaviour, making the control over the vehicle either impossible, or a very difficult task, requiring negative adaptation by the pilot. If manual control is desired in the simulator, then the effects of such a device will require re-learning of the control-related perception and action process by the operator.

With current technology, the third level of fidelity is the highest which is normally achieved, while we must aim to achieve Level 2 for ab-initio zero-flight-time training.

Motion cueing fidelity in the flight simulator

Motion systems are comprised of the motion washout algorithm which transforms the aircraft motions to simulator motions, and the motion cueing mechanism which moves the simulator platform and cockpit[4]. The motion washout algorithms attempt to simulate the onset accelerations directly, while the long-term linear accelerations are synthesized through tilting of the cab and executing the tilt below the perception thresholds[5].

A wide range of washout algorithms are available[6,7], however each performs a transformation on the aircraft accelerations in such a way that the gain and phase of the resulting simulator response is changed with respect to the aircraft input.

With regard to motion platforms, current-day simulators require that the motion system carries large payloads, which include the support frame, visual display system, cockpit, and additional hardware. The high mass moments of inertia can limit the bandwidth of the simulator, and can introduce phase lags in the motion response[8]. Note that these phase lags originate from a number of sources, and not only the motion system. To overcome phase lags, the pilot is required to adapt his control strategy by adding lead compensation. This may not be a problem for the pilot who is sufficiently experienced with flying tasks.

Simulation technology research in SIMONA

The International Centre for Research in Simulation, Motion and Navigation Technologies, SIMONA, is a multi-disciplinary initiative of the Delft University of Technology. Through the cooperation between the Faculty of Aerospace Engineering, the Faculty of Mechanical Engineering and Marine Technology, and the Faculty of Electrical Engineering, SIMONA is developing facilities which will allow top-quality research to investigate the problems described above. The core of SIMONA is a research simulator which has been designed and developed by SIMONA, in cooperation with industrial partners[9]. This device, the SIMONA Research Simulator (SRS) is a tool for investigations into the presentation of environmental and inertial motion stimuli to the pilot, and to assess the transferability of behaviour from the simulator to the flight vehicle. Making valuable contributions to aviation safety is of utmost importance.

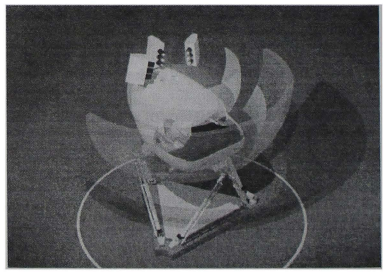


Figure 1 - Integrated SIMONA Research Simulator will house two operators in a re-configurable shell, driven by the 6 DOF hydraulic motion system.

The SRS is shown in Figure 1, in a conceptual form. The motion platform, designed by SIMONA and constructed by the TU Delft

Central Workshop (CW), makes use of six linear hydrostatic actuators, each with an operational stroke of 1.15 m. The SRS motion system forms a closed kinematic chain known as the Stewart Platform, comprised of the base frame, the actuators, and the moving platform. Robust control techniques have been applied to the jacks so that their behaviour represents a force-generator that is less sensitive to load variations[10]. Furthermore, the motion system is controlled by a multi-variable controller which uses six Digital Signal Processors (DSP's) to regulate the platform state. These technologies, primarily developed by the Faculty of Mechanical Engineering and Marine Technology, together with the low inertia moving load, will allow a high inertial motion cueing bandwidth.

The second unique element of the SRS is a multi-vehicle re-configurable cockpit, which is functionally integrated with the upper platform of the motion system. This device, called the "Shuttle", will house up to two pilot stations in a side-by-side seating arrangement. While the environment is generic, SIMONA has particular interest in the simulation requirements of transport aircraft, rotorcraft and road vehicles. The multi-vehicle capability is made possible through the application of a wide-angle projection-based continuous collimating visual display system.

The initial configuration of the interior will represent a twin turbine-engine transport aircraft. Vehicle mathematical models, developed by the Faculty of Aerospace Engineering[11], include a variety of small to large transports. A glass cockpit with four 15-inch programmable EFIS displays will nearly fill the instrument panel, and a complete control stand from the Boeing 777-200 will effectuate the throttles, flaps, speedbrake, and indicate the pitch trim.

The radio navigation simulation in SIMONA represents a step forward in flight simulation, and is the responsibility of the Faculty of Electrical Engineering. The NAVSIM model will incorporate simulations of standard aviation radio navigation signals, GPS, MLS and others, as well as the synthesis of anomalies to their signals. The purpose here is to realistically represent phenomena of particular relevance to training, or to research into navigation systems and displays.

Beyond the vehicle simulation applications, SIMONA is highly interested in the development and validation of models of the human perception process. Due to its high motion cueing bandwidth, and because of its powerful image generation and display system, the SRS is an appropriate tool for investigations into human perception psychology.

The application of advanced light-weight materials in the SIMONA Research Simulator shuttle, and in the visual display system structural hardware, lead to a gross moving payload with a mass just below 3500 kilograms. This includes the on-board simulated aircraft hardware. Furthermore, the favourable mass properties, combined with the powerful hydraulic motion base and its control system, will permit inertial motion cueing with low phase lags over the frequency ranges of interest of flight vehicle control.

The hardware developed by SIMONA requires mathematical models which synthesize the vehicle and command the cueing systems. Since the form of these mathematical models is a focal area of SIMONA research, a flexible software environment is being developed[12]. A critical aspect of research simulation is that new concepts in vehicle simulation and control be tested, without the developer to be a software or programming language expert. The software structure will enable the rapid development of mathematical models, the testing of their performance, and implementation in the real-time environment.

Status of SIMONA

Currently, the SIMONA Research Simulator is undergoing the mechanical integration and test phase. Figures 2, 3 and 4 show the hydraulic motion system during the static calibration tests. Note the temporary steel dummy platform, which kinematically approximates the attachment points of the shuttle, but also allows the mass properties to be changed by attaching steed blocks to the upper flanges. This allows then the identification of the motion base's dynamic response for a variety of payload mass distributions.

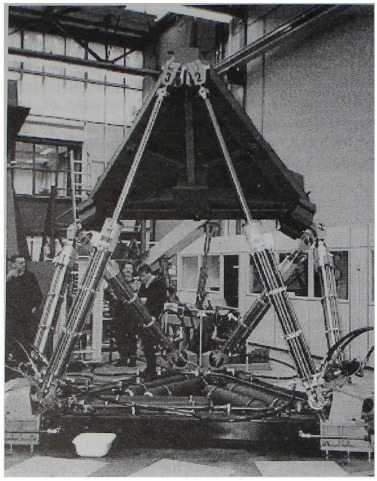


Figure 2 - SRS motion system undergoing kinematic tests Note the temporary variable-mass steel-frame dummy platform

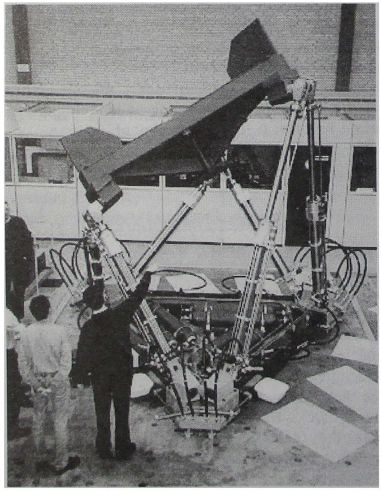


Figure 3 - SRS motion system, with platform in combined pitch-roll attitude

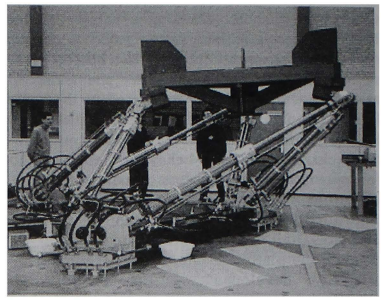


Figure 4 - SRS Motion system, with platform undergoing maximum translational extension, to examine mechanical clearances.

Motion System Static Calibration Tests

One recent effort of SIMONA has been the static calibration of the motion system. During the production and installation of a synergistic 6 DOF Stewart Platform-type motion system, as that of the SIMONA Research Simulator, the positioning accuracy of the platform can be influenced by a number of causes. Manufacturing tolerances, installation errors and leg lengths offsets can cause deviations with respect to the nominal kinematic parameters of the platform. Consequently, if the *nominal* values of these parameters are used within the control software, the resulting pose (i.e. position and orientation) will be *inaccurate*. To ensure positioning accuracy of the platform, one must determine through static calibration the actual values of these parameters, and compensate in the motion software for the errors. The calibration program, carried out in a number of phases, is now described.

Defining the parameters

Figure 5 shows the general arrangement of a Stewart Platform-type motion system. The actuator lengths l_i, and the vectors a_i and b_i of the upper platform and base-frame respectively determine the instantaneous pose (position and orientation) of the platform. Two-degree-of-freedom gimbals are located at the ends of vectors a_i and b_i. In the ideal situation the gimbal points are located on two circles with radii Ar and Br. The pilot position is located somewhere on the moving platform. Under these conditions, a total of 42 dimensional parameters need to be identified in order to determine the positioning accuracy of the platform[13].

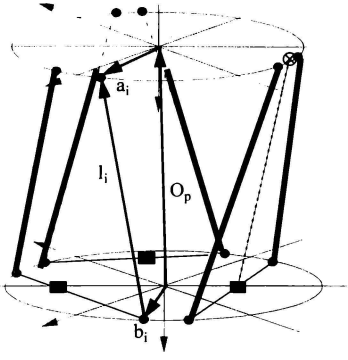


Figure 5 - General arrangement Stewart Platform, including position measurement instruments.

These parameters consist of the x, y and z coordinates of each of the 12 gimbal points in their local reference frame, and the minimum lengths of the actuators.

Sensitivity analysis

The Jacobian matrix maps the platform rates to the actuator rates; in other words, it describes the position accuracy of the platform position and orientation as a function of the positioning accuracy of the actuator lengths. By using Jacobian matrices, the influence of inaccuracies of the kinematic parameters on the platform pose have been analysed[14]. It was found that the gimbal point location errors amplified the platform pose error by an average factor of 1.5. Leg length offsets appeared to have a smaller impact on platform pose error.

Direct geometric measurements

While the actual kinematic gimbal point cannot be measured directly due to the complex mechanical structure of the actuators and gimbal boxes, a number of reference points were however selected. Based on these points, the actual kinematic positions in the local reference frames could be determined. Experimental results showed no deviations larger than two millimeters in the base frame and the actuators, however a deviation of 38 mm was found in the radius of the steel dummy platform. This was isolated by observing a slight pitch when pure x and y translations were commanded. This process is described below.

Indirect measurements with inclinometer

The goal of these measurements was a straightforward check on the accuracy of the gimbal points and the true minimum lengths of the six actuators. By measuring the roll and pitch angles of the moving platform in a number of positions were they should remain zero, some conclusions about the accuracy could be drawn. These angles were measured with an inclinometer, a closed-loop servo accelerometer with high sensitivity in the region about ±1 **g**. The platform was commanded to assume seven different poses, all with commanded roll and pitch angles equal to zero. The ideal leg lengths were calculated using an inverse transformation algorithm, and the actuators were set to the corresponding lengths using the internal Temposonic LVDT's. Figure 6 shows the results obtained using the original kinematic parameters. Note the large deviation when the platform is commanded in pure surge or pure sway motions. After considerable effort, the deviation in the location of the upper platform gimbals, due to a simple manufacturing error, was located.

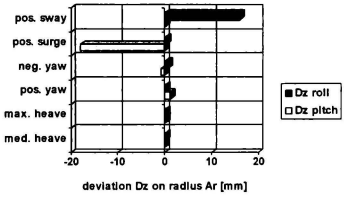


Figure 6 - Results of inclinometer measurement with the original radius Ar

After implementation of the actual radius A, in the motion drive software, the results of the inclinometer measurements improved significantly. The remaining errors, shown in Figure 7, are within reasonable tolerances for a platform of this size.

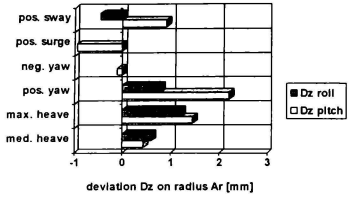


Figure 7 - Results of inclinometer measurement with newly identified radius Ar

Indirect identification with position sensors

a) Indirect tetrahedral pose measurements

Using three position sensors located on the base frame, the position of three arbitrarily chosen points on the dummy platform can be identified[15]. Linear cable-type position encoders, which can measure with an accuracy of 0.05 mm, were used. These units were placed on the base-frame at the mid-point between the lower gimbal boxes. The distance between the encoders can be measured by using the position sensors themselves. The position of the three points can be analytically determined because of the tetrahedral form that is created. Since the three localized points define a surface, the pose of this surface can be calculated with respect to the surface formed by the three encoders.

b) Parameter identification using leg lengths

In 32 independent situations, the platform pose was determined and the leg lengths were traced exactly. Using a least-squares criterion, the 42 aforementioned parameters could be identified. This is accomplished by using two different algorithms[16]. The accuracy of the final results depends largely on the quality of the measured data. This process of exact identification is currently in its final stage.

Hardware status

The SRS shuttle is shown in Figure 8, prior to the completion of the gimbal joints. Note the large machined aluminum blocks which transfer the actuator loads to the composite shell. This structure is built from Twaron®§ and carbon-fiber reinforced polymers.

§ Twaron® is a registered trademark of Akzo-Nobel N.V.

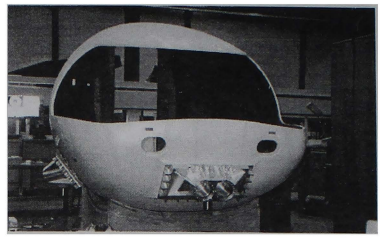


Figure 8 - The SRS integrates the motion platform / cockpit requirements in one hardware element. A frameless window opening and wide interior allow multiple research applications. This unit weighs 817 kg and is built from Twaron® and carbon-fiber laminates.

The mechanical integration will continue through 1997. It is expected that the visual display system, part of which has now been manufactured, will be installed during the middle of 1998. While a complete simulator with visual will be available late in 1998, a large amount of research and development continues. Research with the motion base began in January 1997.

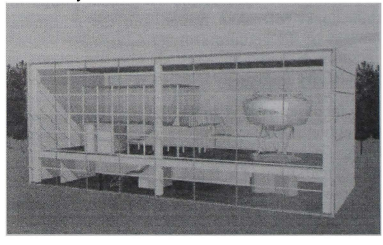


Figure 9 - SIMONA laboratory houses simulator related activities on upper levels and off-line research at ground- level.

SIMONA's activities are centralized about a new dedicated research laboratory, located at the TU Delft. See Figure 9. The ground floor houses the staff, visiting researchers and students directly involved with SIMONA. All simulator operation and development work takes place on the level above. Note that the light-weight simulator is placed on a concrete pedestal at the level of the upper floor.

Future Research Activities

The SIMONA Research Simulator is a generic, flexible tool for a wide range of research which is intended to have a major impact on the application of simulation technologies to training. Additionally, research conducted into vehicle flying qualities in a highly-realistic simulation environment allow the definition of man-machine interfaces of future vehicles. Some of the anticipated projects related to aviation safety are now listed.

Simulation techniques for future training

As mentioned in the introductory remarks, with the increasing replacement of training in flight vehicles through simulator-based training, a number of human perception and action questions need to be addressed. In particular, the effects which the inherent compromises created through simulated flight have on the learning of flying tasks, and the correct execution of responses by the pilot, need to be fully understood. Motion washout filters which reproduce the inertial motions such that the control strategy execution by the pilot remains the same as it would in the vehicle need further development and investigation, before airlines and certification authorities can be satisfied with regard to the training value of the motion simulation.

While the aforementioned addresses the high-end of training technologies, low-cost simulator development represents a majority of industrial activity in training systems. Recurrent training in flight simulators with reduced degrees-of-freedom motion bases is of growing interest, due to the anticipated cost savings which the simplified mechanisms would allow. Three and four-degrees-of-freedom motion bases offer these advantages, however eliminate a portion of the inertial motions. Determining which degrees-of-freedom are necessary for a particular task in a given vehicle category requires off-line analyses, as well as rigorous experimental research in a facility like the SRS.

The visual environment is at least as critical as the whole-body cues provided by the motion system. In the introductory section, the role of the visually-induced motion (known as vection) in conjunction with the inertial motion cues, was introduced. In fact, the perception of visual information requires recognition of the optical field by the observer. Naturally, compromises must often be made: Despite rapid surges in the performance of image generators, it is necessary to understand the requirements of the visual display system for a range of simulator applications. For example, increasing the size (particularly the horizontal field-of-view) of the visual display system increases the cost and the size of this system, however greatly enhances the perception of motion[17].

Research into Human-Machine Interfaces

With the availability of a superior tool for the synthesis of flight vehicles, and the fact that this system's hardware and software will be able to represent a multitude of vehicle types and categories, it will be possible to conduct detailed investigations into the human-vehicle interactions in the flight environment. The mathematical models representing non-linear phenomena, such as aeroelastic behaviour, interaction with the non-stationary environment, and the dynamics of landing gear, will serve to reduce the gap between simulator and flight vehicle.

Current electronic flight displays represent a significant, yet conservative step forward from previous-generation mechanical instruments. Research into the requirements of three-dimensional perspective displays has been carried out by the Faculty of Electrical Engineering[18], and the Faculty of Aerospace Engineering[19]. These display types, which present to the pilot integrated guidance and navigation information through tunnel-in-the-sky symbology, represent a candidate to integrated navigation systems.

Flight control, including fly-by-wire, is a key research topic of the Faculty of Aerospace Engineering. The evaluation of advanced control concepts in a simulated environment allows basic-level clearance of these concepts. The pilot interface to the aircraft control systems takes the form of either side-stick controllers, or standard control columns. Note that these will be interchangeable, and both types of systems will be hydraulically driven. The active loading of the side-stick, in particular, allows research into passive and active controls.

CONCLUSIONS

SIMONA is proceeding toward the completion of its facilities. It is expected that, through international collaboration in scientific and industrial networks, the research conducted in SIMONA and the technologies developed in the process will have a significant impact on aviation safety. Enhancing the training value of simulation technologies will be possible with the utilization of the flexible simulation tools, as will research into future vehicle human-machine interfaces. The capability of a facility like SIMONA depends on the quality of its hardware. Efforts like the identification and correction of the motion platform kinematics are therefore vital to the success of this facility.

SIMONA seeks cooperative ventures with the international community of simulator users, certification agencies, and the simulation industry.

ACKNOWLEDGEMENTS

SIMONA development has been funded primarily by the Delft University of Technology, the Netherlands Ministry of Education, Culture and Science, and industry.

REFERENCES

[1] Hosman, R.J.A.W., "Pilot's perception and control of aircraft motions". Doctoral dissertation, Delft University of Technology, November 1996.

[2] Advani, S.K., Doctoral dissertation, to be published, exp. 1997.

[3] Steen, H.F.A.M. van der, "The application of human orientation and motion perception in the visual and inertial motion control of flight simulators". From Proceedings of CEAS Symposium on Flight Simulation Technologies, "Making it Real", Delft, October 1995.

[4] Martin, E.A., "Force and Motion Cueing Systems". In Flight Simulation Update, American Inst. of Aeronautics and Astronautics, New York, January 1995.

[5] Hosman, R.J.A.W., and Vaart, J.C. van der, "Vestibular models and thresholds of motion perception. Results of tests in a flight simulator". Report LR-265, Faculty of Aerospace Engineering, Delft University of Technology, 1978.

[6] Baarspul, M., "The generation of motion cues on a Six-Degrees-of-Freedom Motion System," Dept. of Aerospace Engineering, Delft University of Technology, Report. LR-248, June 1977.

[7] Ish-Shalom, J., "Design of Optimal Motion for Flight Simulators". Ph.D. dissertation, Massachusetts Institute of Technology, 1982.

[8] Advani, S.K., and Verbeek, R.J., "The Influence of Platform Inertial Properties on Simulator Motion System Performance". AIAA-94-3418-CP. Flight Simulation Technologies Conference, Scottsdale, August, 1994.

[9] Advani, S.K., "The Development of SIMONA: A Simulator Facility for Advanced Research into Simulation Techniques, Motion System Control and Navigation Systems Technologies". AIAA-93-3574-CP. From AIAA Flight Simulation Technologies Conference, Monterey, August, 1993.

[10] Schothorst, G. van, Teerhuis, P.C., Weiden, A.J.J. van der, "Stability Analysis of a Hydraulic Servo-System Including Transmission Line Effects". From Proceedings of International Conference on Automation, Robotics and Computer Vision, Singapore, November 1994.

[11] Baarspul, M., "A Review of Flight Simulation Techniques". In Progress in Aeronautical Science, Royal Aeronautical Society, 1988.

[12] Gool, P.C.A. van, "Software Design for a Reconfigurable Simulator". AIAA Flight Simulation Technologies Conference, San Diego, AIAA-96-3470-CP, August 1996.

[13] Wang J., Masory O., "On the Accuracy of a Stewart Platform - Part 1: The Effect of Manufacturing Tolerance". Proceedings of 1993 IEEE Int. Conf. On Robotics and Automation, pp. 114-120, 1993.

[14] Ropponen T., Arai T., "Accuracy Analysis of a Modified Stewart Platform Manipulator". Proceedings of 1995 IEEE Int. Conf. On Robotics and Automation, pp. 521-525, 1995.

[15] Geng Z.J. and Haynes L.S. (1994).,"A "3-2-1" kinematic configuration of a Stewart Platform and its application to six Degrees of Freedom Pose measurements". Robotics and Computer Integrated Manufacturing, Vol.11, No.1, pp. 23-34, 1994.

[16] Innocenti C. "Algorithms for Kinematic Calibration of fully-parallel Manipulators". Computational Kinematics, Kluwer Academic Publishers, pp. 241-250, 1995.

[17] Steen, H.F.A.M. van der and Hosman, R.J.A.W., "Perception of coherence of visual and vestibular velocity during rotational motion". From 6th IFAC Conference on Man-Machine Systems, Boston, MA, 1995.

[18] Theunissen, E., "Integrated Design of a Man-Machine Interface for 4-D Navigation". Doctoral dissertation, Delft University of Technology, February 1997.

[19] Mulder, M., "Modeling manual control of straight trajectories with a perspective flight-path display". XVth European Annual Conference on Human Decision Making and Manual Control, 1996.

ABSTRACT

The Mishap Analysis Animation Facility (MAAF) has developed over 50 mishap animations, some have been used in flight safety meetings to train aviators. In the animation, actual data drives the instruments and the aircraft model. Animations have the advantage that the mishap can be seen from either the cockpit view, the ground view or the chase plane view. All actions can be presented in slow motion or real time. The MAAF has reconstructed flight paths from ground radar data, AWACS radar data, flight simulator data, Heads Up Display data, and flight manual performance data combined with witness reports. MAAF animations include Defense Mapping Agency terrain data, runways and runway lighting and image mapping.

Flight simulators answer many questions raised by accident investigators, and are often used in the area of aircraft performance under normal configurations, normal power settings, etc.. Flight simulators have been used to generate data in an effort to recreate the mishap. But experience shows that recreation of the exact scenario is very difficult if not impossible.

The author would like to encourage flight simulator development to include input data from the mishap and allow the simulator pilot to overtake the controls at the critical time to see if the accident could have been avoided.

ANIMATION AS AN AID TO MISHAP INVESTIGATION AND TRAINING

Bob Kerr
Chief , USAF Mishap Analysis & Animation Facility
Tinker AFB, Oklahoma

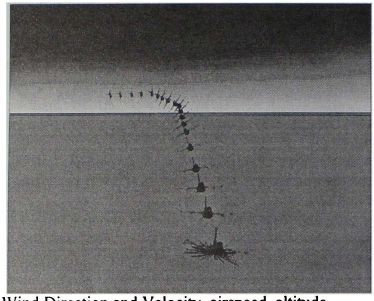
A large USAF airplane is slightly off course. The pilot turns back toward the centerline of the instrument course. It is night with an overcast sky and visibility is zero. The crew is practicing a night terrain following mission. Suddenly the airplane impacts a granite cliff.
The night sky is lit up as the fuel explodes. An airliner flying miles away reports the fireball. How could this happen? What actually occurred? Was the airplane system at fault? How can we prevent this from happening again? These are just a few of the questions facing the investigators. There is not much left of the airplane. What the impact didn't destroy, the fire did. How can investigators develop answers? The key is the Flight Data Recorder or what is often called "the black box". In this case The impact was so great the memory board from the black box was found to have bounced back from the cliff, one-mile. Some of the memory chips were cracked open, but the recent flight data survived on unbroken chips. The author was asked to process the data from the unbroken chips for the mishap board. One bit in one of the data words revealed that the terrain following avionics had been turned off 3 seconds prior to impact. Other chip data indicated that the terrain following avionics had in fact issued a pitch up command. This data saved the USAF a very expensive program to test the terrain following avionics and also provided some of the answers to the investigators. As a result, the USAF funded development of the MAAF at Oklahoma City's Air Logistics Center. Recently, the MAAF has been transferred to the Air Force Safety Center.

In an Air Force Policy letter dated 16 Nov. 1982, the Air Force Chief of Staff directed that Survivable Flight Data Recorders be installed on all new weapon systems. Waivers would require HQ USAF approval. With the advent of modern computer technology, large amounts of data can now be stored in a very small memory chip. Newer aircraft systems store hundreds of parameters, all of which are important to resolve what the crew and the airplane did. Large amounts of data are hard to visualize and time correlate. Fortunately, technology has also evolved to a point where this data can be animated into images on a screen. This allows

This paper is declared a work of the U.S. Government and is not subject to copyright protection in the United States

What is a MAAF animation?

In a MAAF animation, recorded flight data drives the model and the instruments. Reported



Wind Direction and Velocity, airspeed, altitude and heading are used to create the flight path. In the absence of FDR data, flight paths are sometimes constructed by calculating trajectory angles from radar displacement vectors. Radar vectors provide a motion path as a function of time which, when differentiated twice, produce velocity and acceleration functions. Using vector analysis, these functions can be resolved into their longitudinal, normal, and transverse components (with respect to the aircraft) which in turn can be used to calculate the pitch, roll and heading angles of the trajectory. The actual aircraft attitude will differ from this tangential attitude by the angle of attack and slip angles. Since angle of attack and slip angle are unknown, the tangential attitude angles provide the closest approximation to the aircraft attitude; and, in practical application, produce realistic aircraft attitudes in animations using radar data.

The MAAF has been processing mishap data and creating animations of USAF mishaps since Sept 94. During that time the MAAF has created over 50 animations. Some have been used by the Air Force Safety Center to develop training videos to help increase Flight Safety. If a picture is worth a thousand words, an animation of a mishap is worth ten thousand words. Since many of the lder USAF aircraft do not have Flight Data Recorders, the MAAF has had to reconstruct flight paths from ground radar data, witness reports, Heads Up Displays on other aircraft, flight simulator output data, and analytical aircraft performance data. Whatever the source, animations include an instrument panel displaying the data used to drive the airplane model. When available, video animations sent to the mishap board include the cockpit voice recordings. The MAAF can do a spectral analysis of the cockpit recording to isolate noise frequencies of interest to the mishap board, and can use digital filters, and remove constant noise sources such as AC hum, engine, environmental systems, etc. thus enhancing the voice recording. Recently the MAAF has included afterburner plumes, birds during bird strike scenarios, cargo drop chutes and cargo drops, canopy ejection Defense Mapping Agency (DMA) terrain elevation data and picture image mapping. The animation videos are used by the Mishap Board Presidents to brief their chain of command.

The MAAF can extract data from FDR's down to chip level for many aircraft. The MAAF has developed data extraction capability for tape type FDR's such as that found on the cargo and commercial type aircraft. Most commercial aircraft and most large cargo military aircraft use flight data recorders that have the data written to magnetic tape. The data is sometimes thought to be invalid because of check sum errors. These errors are caused by wow, flutter, dirty heads, bad tape quality, etc. The analysis software allows the MAAF to look at individual bits and wave patterns to correct missing or improperly



interpreted bit data. Thus critical mishap data can often be salvaged.

The MAAF has also developed the capability to extract data from the Voice and Data Recorder (VADR) used on many helicopters.
The MAAF funded development, of the terrain and image mapping software, and has provided this software module free to accident investigators from several countries who are involved in both military and commercial

accident investigations. The Canadian Transportation Safety Board and Software Kinetics Ltd. wrote the main body of the animation software in Canada.

How does an Animation differ from a Flight Simulator?

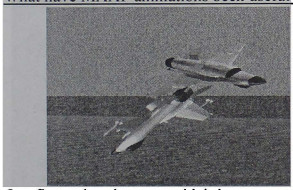
Flight Simulator trainers can answer many questions raised by accident investigators and are often used in the area of aircraft performance under normal configurations, normal power settings, etc. But the problem is that drag curves have not been developed for flight simulators for every unlikely configuration that may occur in a mishap. Flight Simulators are good for answering the question of how could the pilot have gotten himself in this situation. Experience shows the re-creation of the exact scenario is very difficult if not impossible in some cases. Usually this is true where an untested configuration is part of the mishap. For example, if part of the airplane has structural damage, or flight controls are damaged.

In animation, actual aircraft attitude, and performance is taken from the flight data recorder and displayed in real time. Unusual attitudes and configurations are no problem for animation if they are within the limitations of the attitude sensors of the flight data recorder.

Sometimes Flight Data Recorders do not work properly and important data is lost. In these cases, flight simulators are used to try to recreate the mishap scenario. Highly trained pilots sometimes have trouble trying to exactly simulate the mishap pilots actions. He may rotate to soon on take off, miss a required airspeed or altitude, flare to soon or to late on landing, etc. so the missing data cannot be accurately determined.

Flight simulators usually have a limited number of terrain scenes, and runway headings. Photos of the mishap area can be image mapped into the MAAF animation. Tactical Pilotage Charts (TCPs) can also be image mapped into the MAAF animation. The MAAF can place terrain elevation data from any part of the world in its animations. Runway headings, runway slope, length, width, and runway lighting can be tailored to the mishap airport.

What have MAAF animations been useful for?



Our first animation was midair between two F-16 fighters engaged in combat maneuvers. We received data from both flight data recorders, created separate flight paths and accurately displayed the resulting midair collision. The MAAF has done animations with as many as seven aircraft, creating flight paths that accurately displayed the resulting mishap. The main benefit of the MAAF animation is the presentation of a large amount of flight data that is all time sequenced. Imagine trying to look at listings or plots of flight data from seven different aircraft and determining aircraft attitudes and relative positions of each aircraft

What did the pilot see? Cockpit views from the pilots design eye point can be displayed with other aircraft and terrain within or without his field of view. Heads Up Display (HUD) can be added to animations with displayed data driven by flight data recordings.

How can animations be used as aid to training?

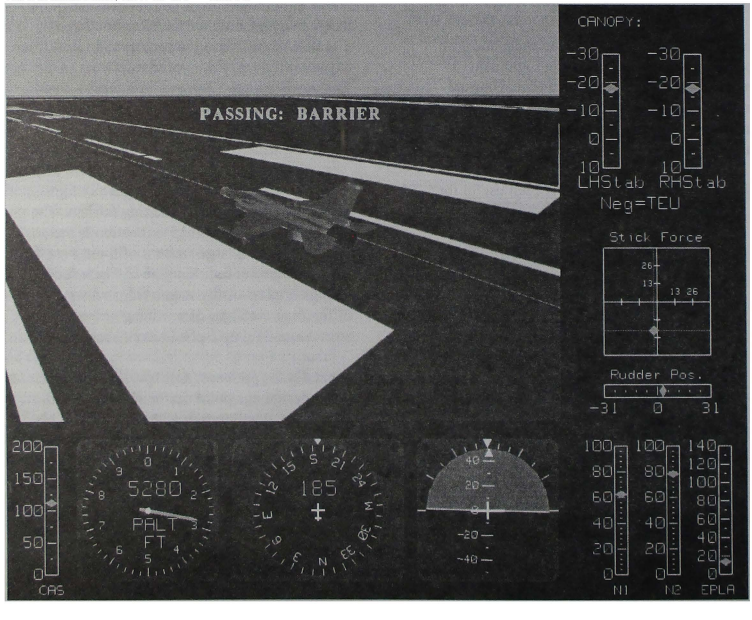
When cockpit voice data is time sequenced with the animation, they have shown the need for better cockpit communications. When pilots are working together in the cockpit as a team, misunderstandings can be fatal. Communications between ATC and the cockpit are also misunderstood. One difference between "normal simulator" simulations and the real world is that

often ATC communications are interrupted by other transmissions. CVR recordings have revealed that sometimes pilots did not understand ATC and instead of asking for a repeat on a busy channel, simply discuss it among themselves. When this practice can be shown to be fatal, aviators will insure they understood the communication.

MAAF animations have shown aircraft to be well off course in an approach meaning there were human errors in interpretation of cockpit data, unacceptable navigation procedures and so on. MAAF animations include the pertinent instrument panel data and when the animation is run real time along with cockpit voice recordings it becomes apparent when the flight crew has overlooked important cockpit data. The question as to why was the data overlooked has to be addressed in training, in location and visibility of instrumentation, and in crew duties.

Lessons learned from actual mishap data, is the best teacher since most accidents are the result of a number of errors which when combined put the flight crew in an unrecoverable position. As such, animation has proved to be a valuable tool in training aviators.

The author would like to encourage flight simulator development to include the ability of a flight simulator to input data from the mishap flight data recorder and allow the simulator pilot to overtake the controls at the critical time to see if the accident could have been avoided. This would also provide a good tool to improve cockpit design, if the resulting scenario shows that the mishap was caused by hard to read or see data. Engineers could study performance improvements to offset the conditions that caused the mishap.



DEVELOPMENT OF ATC SIMULATION FACILITY

Kakuichi SHIOMI, Hiroki SATO, Yutaka FUKUDA,
Kota KAGEYAMA, Toshihiro HIWADA* and Kazuo SHIMA*

Electronic Navigation Research Institute, Ministry of Transport
6-38-1, Shinkawa, Mitaka, Tokyo 181, JAPAN
(*)CRC Research Institute, Inc.
2-7-5, Minamisuna, Koto-ku, Tokyo 136, JAPAN

Abstract

The Electronic Navigation Research Institute is now developing an integrated air traffic control (ATC) simulation facility using virtual reality technology.
The facility currently has an ATC tower simulator and a flight simulator which are connected to each other. The flight simulator pilot operates his aircraft based on instructions from the air traffic controller who is participating in the ATC tower simulation, and the controller in turn conducts his activities while watching moving images of the aircraft.
As of 1997, the ATC tower simulator's visual system, consisting of a 360-degree screen with a 15-meter diameter and eight projectors, is the biggest virtual reality simulation system in Japan.
This facility is expected to aid in areas such as: (1) the development of instruments used by controllers, (2) the development of ATC methods that enable efficient ATC operation, (3) research communication between air and ground by voice and data-link, and (4) research from various viewpoints on airports yet to be built.

1. Introduction

In 1991, the Electronic Navigation Research Institute (ENRI) started the development of a computer-oriented air traffic control (ATC) data and information processing system for ATC tower controllers (TDP: Tower Data Processor).
In 1993, ENRI also started the construction of a virtual reality simulation facility (FaVRS) for the evaluation of the system mentioned above. Figure 2 shows the structure of the ATC simulation system installed in FaVRS in its present condition as of 1997.
The construction of FaVRS represents ENRI's first effort to develop an ATC tower simulator based on virtual reality technology. The ATC tower simulator consists of an eight-channel visual system and a host computer as a simulation server. The visual system which provides the view to the controller like that shown in Fig. 1 consists of an eight-channel image generator, eight image projectors, and a 360-degree curved screen with 15-meter diameter and 5.7-meter height.
In 1994, a flight simulator was added by improving the image generator and the simulation server. The image generator was expanded to an 11-channel system, and



Figure 1. ATC tower simulation and VTOL controlled by flight simulator

improved software for processing of simulation scenarios was installed in the simulation server. In the flight simulator, the out-the-window view from the cockpit is provided by three channels of the 11-channel image generator.

Through the above-mentioned improvement, FaVRS acquired the function of two coupled virtual reality spaces. In FaVRS, the flight simulator pilot operates his aircraft according to instructions coming from the ATC controller who is participating in the ATC tower simulation, and the controller in turn conducts his activities while watching the moving image of the aircraft.

In 1995, a voice processing system was developed to evaluate TDP in FaVRS. The realization of the voice processing system for the ATC tower controller had been necessary to carry out the ATC tower simulation. In the ATC tower simulation, there are a large number of mobile aircraft on the airport surface which must be moved according to instructions coming from the ATC controller. One flight simulator is therefore not enough to carry out a full airport simulation. The voice processing system, which consists of a voice recognizer and a voice synthesizer, allows aircraft on the airport surface to move automatically based on the controller's instructions and provides voice replies from virtual pilots to the controller.

In 1996, a fast-time (faster than real-time) simulator was installed to generate simulation scenarios for real-time simulations. The fast-time simulator's kernel is software called TAAM from The Preston Group in Australia, and original software developed by ENRI converts TAAM's output into scenarios for the real-time simulation software called STAGE which is provided by Virtual Prototypes, Inc. of Canada.

The functions of FaVRS were realized by gathering many commercial-off-the-shelf (COTS) software packages as well as software developed by ENRI.

Figure 3 is a picture of FaVRS, where the ATC tower simulator's 360-degree curved screen is installed in the cylindrical part of the building.

Table 1 shows the list of major hardware products installed within the facility, and Table 2 shows the software packages which are used by FaVRS.

2. Simulation Server / Scenario Generator

The scenario processor processes the simulation scenario in tandem with the ATC tower simulator and the flight simulator and controls positions of aircraft and ground vehicles. Depending on the scenario, the scenario processor controls various weather effects such as wind direction, wind strength, visibility, clouds, rain, and snow. The scenario processor sends aircraft position data to a radar image processor which displays a radar image like that provided by a Secondary Surveillance Radar. Events like accidents and near-misses can also be generated within the scenario. The scenario processor controls more than 512 aircraft and ground vehicles simultaneously at a maximum refresh rate of 120 Hz. The maximum geographical area which is supported by the simulation processor is 2,000km X 2,000km per process.

The scenario generator generates simulation scenarios that will be processed by the scenario processor based on flight plans, an airspace, and airport maps.

ENRI is now preparing databases for the simulation of Haneda Airport and Narita Airport.

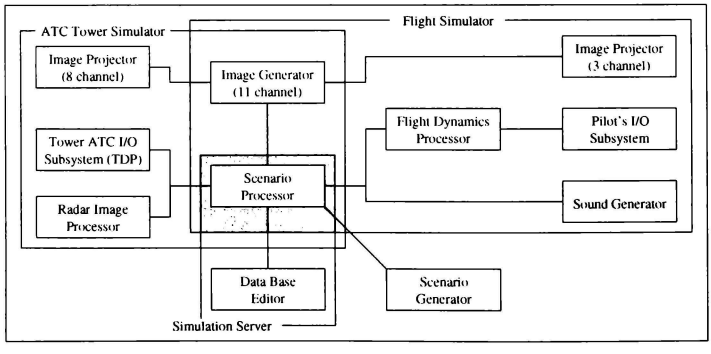


Figure 2. System diagram of the virtual reality simulation facility

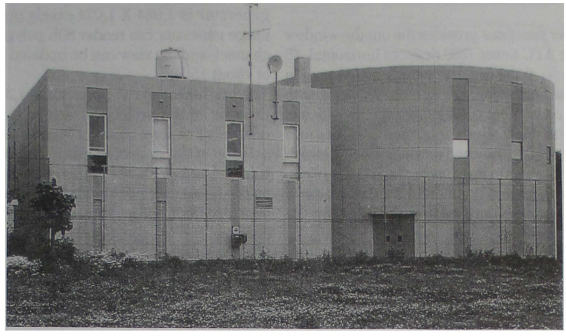


Figure 3. Virtual reality simulation facility of ENRI

[ATC tower simulator and a flight simulator]
Image Generator: Ball-944 11 channel (Ball Imaging Products)
Scenario Server: Onyx VTX (SGI)
Flight Dynamics Computer: Onyx RE2 (SGI)
Cockpit Display Subsystem: Indy x 3 (SGI)
Radar Image Display Subsystem: METHEUS Omega 5700 Display System (Metheus Corp.)
[TDP]
Local Scenario Processor: SPARCstation 20 Multi-CPU x 1 (Sun Microsystems)
Controller's I/O Subsystem: SPARCstation 20 Multi-Monitor Multi-CPU x 4 (Sun Microsystems)
Voice Recognizer: DS200 x 1, PE-200 x 2 (Speech Systems, Inc.)
Voice Synthesizer: DEC-talk x 3 (DEC)
[Other Hardwares]
File Server: Auspex 7000 (Auspex Systems, Inc.)
Additional Processors: CRAY EL98 (SGI), CRAY APP-728 (SGI)
Network Router: Power Hub 7000 (FORE)

Table 1 Hardware

[ATC tower simulator and a flight simulator]
Image Generator Animation: RSR (Ball Imaging Products)
Scenario Processing and Data Base Editor: STAGE (Virtual Prototypes, Inc.)
GUI Build and Management: VAPS/CCG (Virtual Prototypes, Inc.)
Fixed Wing Flight Dynamics Processing: FLSIM (Virtual Prototypes, Inc.)
Helicopter Flight Dynamics Processing: FLIGHTLAB (Advanced Rotorcraft Technology, Inc.)
Sound Generator: AudioWorks (Paradigm Simulation, Inc.)
Local Animator: VisionWorks (Paradigm Simulation, Inc.)
Local Animator: VEGA (Paradigm Simulation, Inc.)
2D Back-Plane Graphics Mapping Software: (ENRI Original Software)
[TDP]
Local Scenario Processing: (ENRI Original Software)
ATC Tower Controller's I/O Sub-System Software: (ENRI Original Software)
AI Process Management: G2 (Gensym Corp.)
[Other Software]
3D Graphics Modeler: MultiGen (MultiGen, Inc.)
Digital Flight Data Recorder Analyzer: ADRAS (AliedSignal, Inc.)
Fast-Time Simulator: TAAM (The Preston Group)

Table 2 Software

3. ATC Tower Simulator

The ATC tower simulator provides the out-the-window view from the ATC tower (360 degrees horizontal, 45 degrees vertical) along with the sound environment in order to create an atmosphere as close as possible to that of actual controllers conducting their activities in the tower. The controller's interface consists of a computerized ATC display and an audio system which enables the controller to give instructions to the pilots. The active aircraft are controlled by the scenario processor and move according to directions from the ATC tower controller.

The out-the-window view from the ATC tower is generated by eight channels of the 11-channel image generator and projected on the 360-degree curved screen illustrated in Fig. 4 and Fig. 5 by eight projectors (Marquee 9000 from Electrohome).
The resolution of the image generated by the image generator is 1,024 X 1,024 pixels per channel. The image generator can render 80K polygons/second per channel, and the view can be updated at a refresh rate of about 15 Hz.
The brightness of the projector is 1,350 lumens, and the gain of the projection screen is about 1.3. The relatively low screen gain of 1.3 was chosen to prevent halation of the projected image and to realize higher contrast. In FaVRS, the edges of the images projected on the screen are not blended so as not to reduce the total resolution and brightness of the simulator's visual displays.

The ATC controller who takes part in the ATC tower simulation sees a view like that shown in Fig. 1 from the stage installed in the center of the screen. Fig. 6 and Fig. 7 show the layout and setup of the ATC tower simulator.

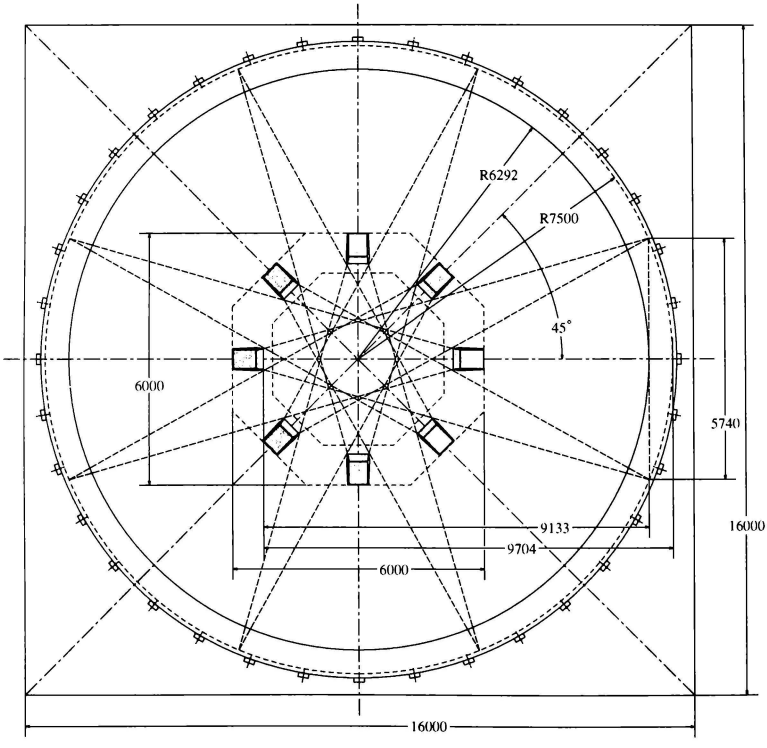


Figure 4. Dimensions of the 360-degree curved screen

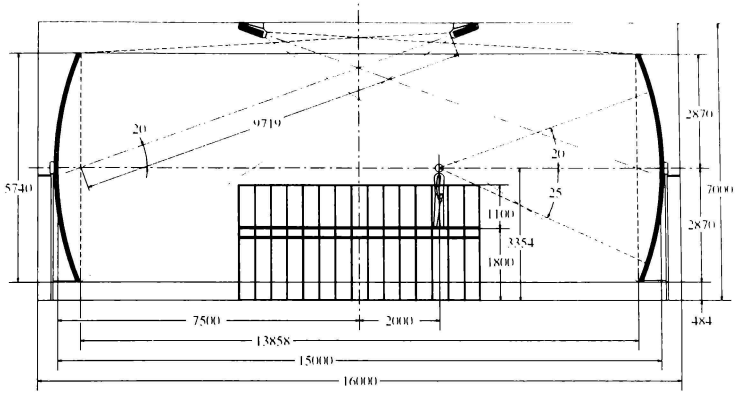


Figure 5. Dimensions of the 360-degree curved screen and the stage

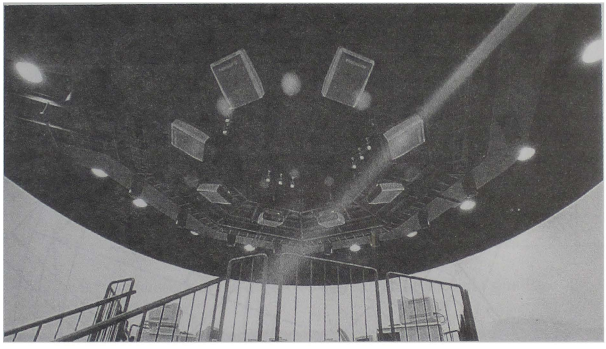

Figure 6. Layout of 8 image projectors


Figure 7. Stage for ATC tower simulation

4. Flight Simulator

The flight simulator provides the out-the-window view from the aircraft cockpit (max. 180 degrees horizontal, max. 60 degrees vertical) and simulation of engine noise while the pilot receives and responds to instructions from the ATC tower. In typical operation, the field of view provided by the simulator is 150 degrees horizontal by 50 degrees vertical.

The view from the cockpit is generated by three channels of the 11-channel image generator and is projected on a 150-degree curved screen illustrated in Fig. 8 and Fig. 9 by three projectors (Marquee 8000-3D from Electrohome). The resolution of the image is the same as that of the ATC tower simulator. Since the common image generator generates images of both the ATC tower simulator and the flight simulator, both images can be completely synchronized.

Figure 10 is a picture of an example of flight simulation.

Figures 11 and 12 show the layout and setup of the flight simulator. The flight simulator has three control sticks, three levers, and two pairs of rudder pedals to enable simulation of both fixed-wing airplanes and helicopters. The responsiveness of each type of aircraft is realistically computed via aerodynamic simulation software. In the current configuration, the aerodynamics of the A320 (Airbus Industries), B747-400 (Boeing), Bell 412 (Bell Helicopter), and V-22 Osprey (Bell/Boeing) can be simulated.

Cockpit instruments are virtually generated on CRT displays, and touch panels are installed to enable control of the cockpit instruments and input of system

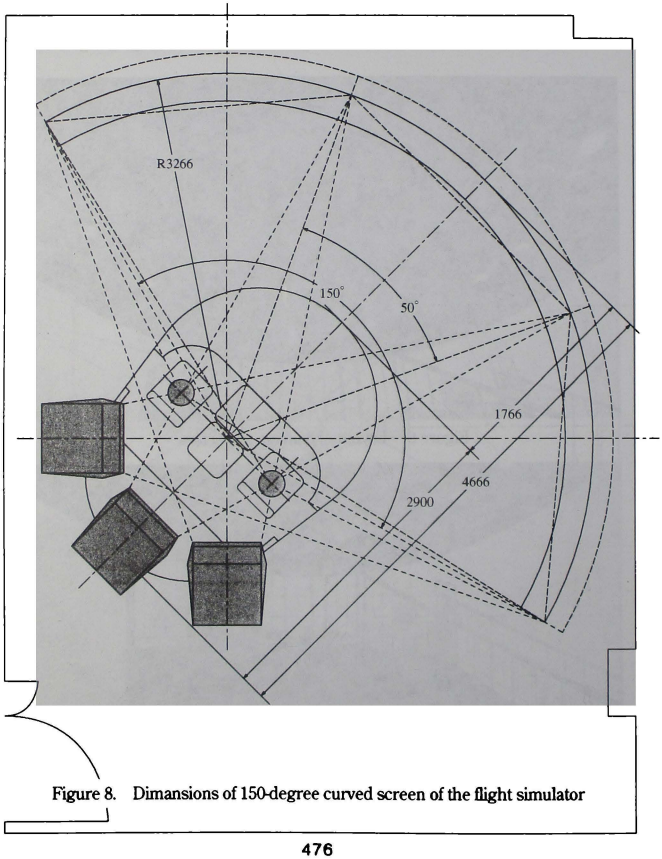


Figure 8. Dimansions of 150-degree curved screen of the flight simulator

management data. Figure 13 shows one example of cockpit display design.

The flight simulator also contains a Flight Management System (FMS) which enables it to simulate autopilot flight, instrument flight, and futuristic data-link-based ATC communication methods.

In addition, the flight simulator was designed to be able to replay data obtained from the flight data recorder of an aircraft in order to investigate the causes of accident or incident cases.

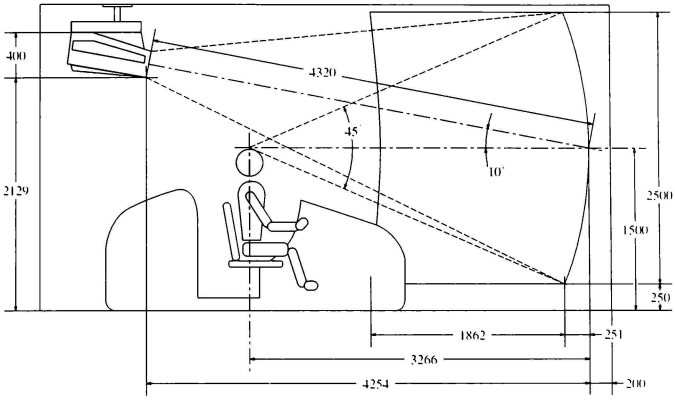


Figure 9. Dimensions of 150-degree curved screen of the flight simulator - 2

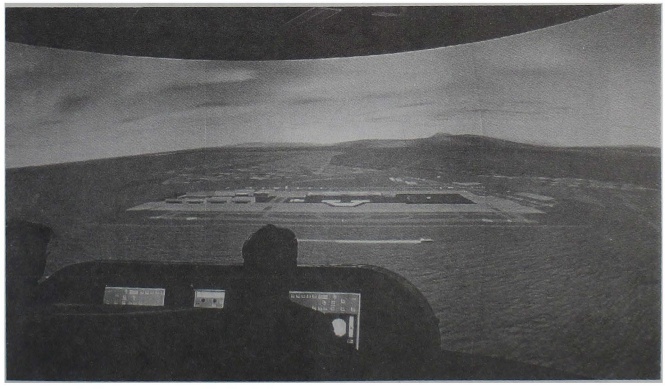


Figure 10. Flight simulation

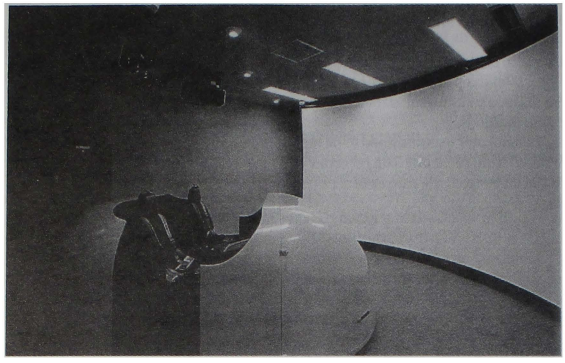


Figure 11. Layout of flight simulator

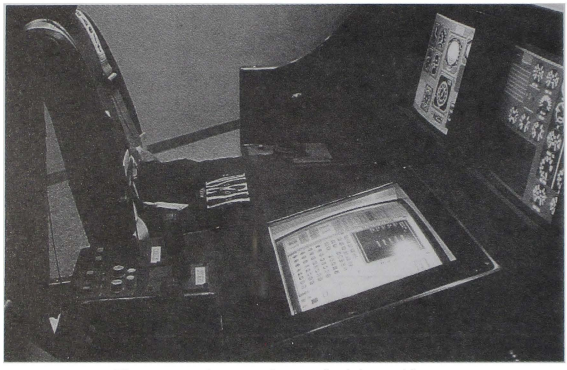


Figure 12. Layout of control sticks and levers

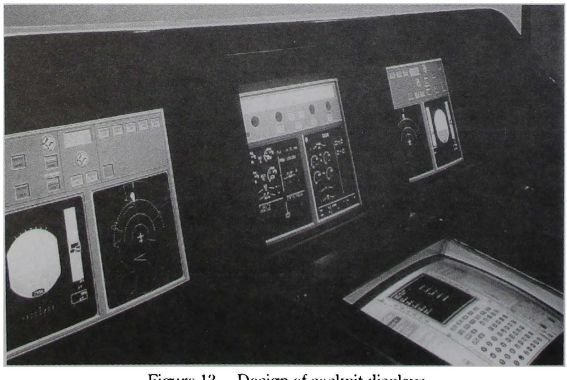


Figure 13. Design of cockpit displays

5. ATC Tower Controller's I/O Subsystem

The ATC tower controller's I/O subsystem has been developed by improving the TDP system. The TDP system had been designed and developed as a prototype system which would be able to enhance the tasks of ATC tower controllers at Tokyo's Haneda Airport. The required functions of the TDP system are as follows:

(1) Front-end console of a FDP (Flight Data Processor: flight plan management system).
(2) Information exchanger among ATC tower controllers.
(3) Controller-Pilot data link communication console in an ATN (Aeronautical Telecommunication Network) paradigm.

The ATC tower controller's I/O subsystem was developed by adding an ATC tower's local scenario processor to the TDP system and creating a virtual voice communication environment. Figures 14 and 15 show the structure of the ATC tower controller's I/O subsystem.

The local scenario processor manages four ATC tower controllers' I/O subsystems (Flight Data, Clearance Delivery, Ground Control and Local Control) and sends ATC event data from each subsystem to the central scenario processor (described as the "scenario processor" in Fig. 14).

In the virtual voice communications environment, the ATC tower controller is able to simulate voice communications with pilots by using a head set. When the ATC tower controller speaks an ATC instruction into his microphone, the voice recognizer converts the voice instruction into an ATC event and sends it to the ATC tower controller's data processor. The ATC tower controller's data processor processes the ATC event as control data for the particular aircraft to be controlled and generates a reply message from the virtual pilot. The generated message is converted to a voice sound by the voice synthesizer and is provided as an audible voice to the ATC tower controller. The aircraft control data processed by the ATC tower controller's data processor is sent to the ATC local scenario processor and is delivered to other ATC tower controller's data processors as well as to the central scenario processor.

In the above-mentioned virtual voice communications environment, a type of artificial intelligence (AI)

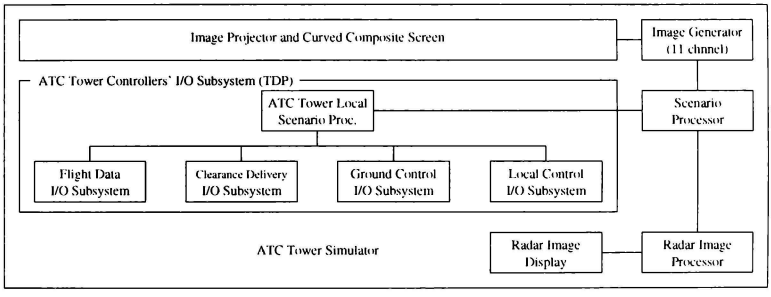


Figure 14. Diagram of ATC tower controller's interface subsystem

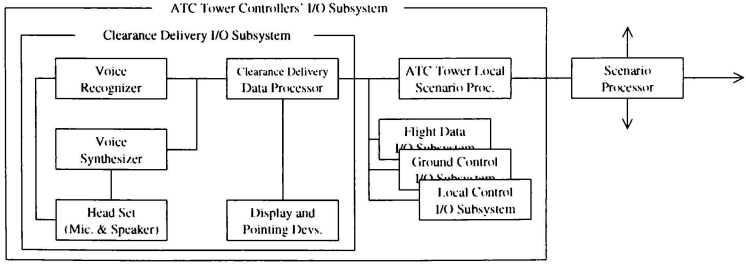


Figure 15. Example: diagram of clearance delivery's interface subsystem

processing system is used to increase the probability of correct voice recognition. Depending on the situation of the controlled aircraft, the AI system makes a list of ATC commands that have a possibility of being spoken by the ATC controller and predicts the instruction which the ATC tower controller intends to command to the pilot.

In Fig. 16, the ATC tower controller who handles clearance delivery interacts with the simulated voice communication.

6. Future Plans

ENRI is now progressing in its development of a terminal airspace (IFR: Instrument Flight Rule) ATC simulator. In its present configuration, two radar image displays have been installed in FaVRS which display the terminal area of the airport simulated by the ATC tower simulator.
An IFR controller's interface system similar to that of the tower controller is now under development.

ENRI and CRC Research Institute, Inc. are now planning to carry out experiments which will couple multiple flight simulators installed at remote locations.

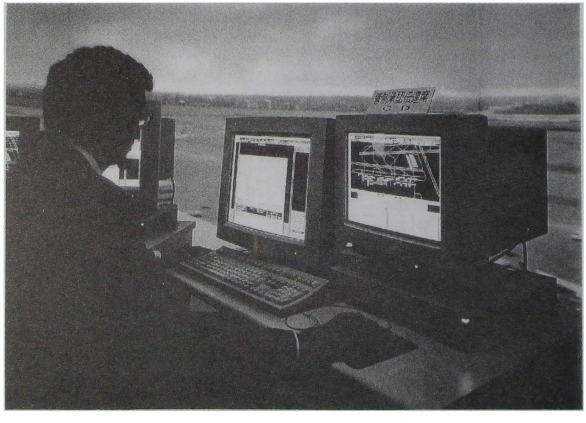


Figure 16. ATC tower controller's interface subsystem

Information

URL: http://www.enri.go.jp
e-mail: shiomi@enri.go.jp

AIAA-97-3728

PC-Based Simulation of the F-16/MATV

Matthew J. O'Rourke*, John N. Ralston†, Jeffery W. Bell‡
Bihrle Applied Research, Inc.
Hampton, Virginia 23666

Stanley F. Lash§
Wright Laboratory
Flight Dynamics Directorate, Flight Control Division
WPAFB, OH 45433

Summary

The Wright Laboratory (WL) F-16 simulation model has been re-hosted on a personal computer running a unique six-degree-of-freedom simulation environment. This model contains over 1000 two-dimensional aerodynamic data tables and a complex flight control system (FCS) modeled by thousands of lines of FORTRAN computer code. The FCS model can emulate several different configurations including a standard Block 40 F-16 and the F-16 Variable In-Flight Stability Aircraft (VISTA) with Multi-Axis Thrust Vectoring (MATV) in numerous modes. In addition to re-hosting the WL F-16 model, the high-angle-of-attack database was extended to include nonlinear effects at moderate to high angles of attack. These additional data greatly improve the ability of the simulation model to predict the in-flight maneuvering characteristics of the F-16 with active MATV.

Introduction

Bihrle Applied Research (BAR) has been conducting simulation studies of stalled, departed and spinning flight of fighter aircraft for over twenty years. To facilitate the modeling and simulation of the complex dynamic behavior encountered during these and other extreme flight conditions, BAR has developed, D-SIX, a six-degree-of-freedom simulation environment for the PC. One distinguishing feature of this software is the unique capability to rapidly re-host aerodynamic databases, flight control system (FCS) code and/or other complex computer code which model subsystems such as weapons, sensors and mission equipment. Due to the complex nature of the computer programs which model these systems, it is important that as much code as possible be reused when a simulation model is re-hosted in another operating environment. This is done not only to minimize the level of effort required, but also to ensure that the accuracy of the model is preserved.

One such simulation model which exists is the USAF Wright Laboratory (WL) F-16/MATV simulation model[1], operated by the Flight Dynamics Directorate, which includes an accurate representation of both the Lockheed Martin F-16 aerodynamic database and FCS. The FCS can be configured to model both the standard Block 40 F-16C and the Variable-Stability In-Flight Simulator Test Aircraft (VISTA) with Multi-Axis Thrust Vectoring (MATV), shown in Figure 1. Each configuration modeled by the FCS, in most cases, employs the identical code used in

* Engineer
† Engineering Manager
‡ Engineer
§ Principal Scientist, Control Dynamics Branch

Copyright © 1997 by the American Institute of Aeronautics and Astronautics, Inc. All rights reserved.

the Lockheed Martin simulation and has been validated to match throughout the entire flight envelope.

In a recent study performed by BAR, a new high-angle-of-attack database was developed to more accurately model the complex aerodynamic characteristics displayed by the F-16/VISTA while flight testing the MATV nozzle.[2] A detailed description of this database is presented in Reference 2. To build upon this effort, BAR has been tasked to re-host the previously mentioned F-16/MATV simulation operated by the Flight Dynamics Directorate at WL using the D-SIX simulation environment, while simultaneously updating the high-angle-of-attack database with that described in Reference 2. This paper will address the differences between the WL F-16/VISTA high-angle-of-attack database and the high-angle-of-attack database assembled by BAR. In addition, experience and lessons learned from re-hosting the FCS will be discussed. Finally, the re-hosted simulation will be validated against the WL simulation as well as flight test.

Brief D-SIX Simulation Environment Description

D-SIX is a MS Windows-based 6-degree-of-freedom simulation environment designed to address issues associated with many of the current complex engineering simulators. Coding requirements are minimized with the extensive use of graphical interface for all simulation functions such as multidimensional table-look-up, flexible plotting capabilities, and data editing. Real-time operation coupled with a low-cost, accessible control interface and visual output permits timely pilot-in-the-loop evaluations of FCS and aerodynamic characteristics. Existing modules from other simulation codes can be incorporated through Dynamic Link Libraries to avoid extensive re-coding and re-validation. Additionally, many tools associated with flight test data are incorporated to allow dynamic, real-time trajectory replay and flight parameter extraction for model validation and refinement. The project-based setup permits simple exchange of simulation models between users, and any capability upgrades in the D-SIX simulation environment is automatically propagated to all existing models. .

Aerodynamic Database Structure and Upgrade

Typical of most current high-fidelity flight simulations, the WL F-16/MATV model provides an accurate representation of controlled, maneuvering flight below maximum lift. However, like most highly-developed fighter simulations, extensive portions of the database have been linearized. In particular, some control deflections such as aileron and rudder have been linearized with deflection, which, for the F-16, are known to be nonlinear at moderate-to-high angles of attack. In addition, no dependency of control effectiveness or basic airframe pitching moment on sideslip is modeled. Basic airframe lateral-directional characteristics of the WL F-16/MATV simulation database are modeled with a linear dependence on sideslip. Reference 2 indicates that a highly nonlinear lateral-directional database accounts for some of the unexpected lateral-directional flight dynamics encountered at high angles of attack during MATV flight testing.

Although the effort described in Reference 2 produced a high-angle-of-attack database, its structure remained significantly different from that of the WL F-16/MATV data structure and included no high-speed aerodynamic effects. Therefore, the challenge existed to incorporate the BAR-developed high-angle-of-attack database with the existing low-angle-of-attack and high-speed database of the WL F-16/MATV simulation without compromising the integrity of either. In addition, to avoid discontinuities as the simulation transitions from the low-angle-of-attack to the high-angle-of-attack database, it is was necessary to include the entire low-speed database in one data set, modeling the Mach and altitude effects as increments taken from the WL database to that unified data set. However, this required that the low-angle-of-attack data, included in the BAR database and used for the low-speed portion of the unified data set, match the low-angle-of attack and low Mach data of the WL F-16/MATV database. This was done to ensure that the Mach and altitude increments yield values identical to the WL F-16/MATV database when added to the unified low-speed data set.

The changes to the database were made in two basic classifications: those to the basic airframe data set and those to the control increments. As stated previously, the basic airframe data were taken from the low-speed dataset in Reference 2 and the WL data were incremented from that (Mach=0.2) data to create basic airframe increments to all six force and moment coefficients as a function of Mach and altitude. The control increments were treated differently, however, due to the very different model structures of the nonlinear control data and the WL control data. To compensate for this, the WL control data were converted to multipliers on the low speed data as a function of Mach and altitude. This allows the full nonlinear control data to be used with the control power

augmentations / reductions with Mach and altitude properly modeled. Great care was taken to ensure that the multipliers are realistically represented when low-speed data values are small.

Flight Control System Re-Host

Both the WL F-16/MATV simulation and the D-SIX simulation environment contain sections that perform identical calculations such as equation of motion integration, stick inputs, coordinate transformation, sensor modeling, etc. However, the FCS is a function of many of the quantities calculated in the these sections of code external to it. Therefore, the external variables which are input to the FCS code in the WL F-16/MATV simulation were identified and mapped to the appropriate D-SIX variables so that the FCS could fit seamlessly into the D-SIX environment. Using code development environments such as Microsoft Visual C++, the FCS can be compiled in its native language (in this case FORTRAN) and linked directly to D-SIX in the form of a dynamic link library.

Correlation With Simulation Check Cases

In order to ensure that the data and FCS have been implemented properly, validation runs have been made in various stages. The first were made in comparison to check cases generated using the WL F-16/MATV simulation. This was done using the original database to ensure that the FCS was transported to D-SIX accurately. The validation runs were performed simply by driving the re-hosted simulation model with the trimmed conditions from the WL simulation output and the control stick time histories. Figure 2 shows a maximum roll input for the Block 40 F-16. Figure 3 shows a roll doublet at 35° α with MATV on. Comparison of both time histories shows good agreement, especially considering that small differences in the high angle of attack database exist between the current WL model and the one used to generate the check cases.

Correlation with Flight Test Data

Once it had been established that the WL F-16/MATV simulation was accurately transported into D-SIX, the BAR high-angle-of-attack database was implemented as discussed previously. To ensure that this database was more representative of the high-angle-of-attack flight dynamic characteristics, the Overdrive feature of D-SIX was used to extract the force and moment coefficients from flight test data gathered during flight testing of the MATV. These data were compared to predictions obtained by running the flight test state parameters and control deflections with the new, fully non-linear database. A more detailed description of the Overdrive feature is presented in Reference 2. Figure 4 provides a flow chart which summarizes the Overdrive function and process. Although this phase of database validation was completed for the high-angle-of-attack database alone in Reference 2, the database validation time histories presented in the this paper use the unified data set described previously. Two examples of the comparison of the total flight-extracted rolling moment coefficient to that predicted by the updated database are presented in Figures 5 and 6 using a flight of the F-16 VISTA/MATV executing a 35° α roll . Figure 5 shows the comparison using the mostly-linear data set in the current WL model, and Figure 6 shows the comparison using the fully non-linear model. Not only does the comparison show that the nonlinear data set matches much better, but the following section will show how these differences are amplified when executing actual maneuvers with the six-degree-of-freedom simulation models. For a more detailed description of the process of database validation with flight test, please refer to Reference 4

Comparison of Simulation Models

Initially, a simple comparison of the current WL model and the fully-nonlinear model was done using a 360° roll at the angle-of-attack limiter with the F-16 in MATV mode, but with the limiter on. Figure 7 shows the time histories for the current WL and nonlinear models. Both models are trimmed at identical states and are driven with the same stick inputs. The sideslip excursion of the nonlinear model is slightly higher and the maximum roll rate is about 10 deg/sec lower than that of the current WL model. However, the two time histories represent very similar responses (unfortunately, at the time this study was completed, not flight test data were available for this maneuver). It now remains to compare the two models in the high angle of attack region where the flight dynamic characteristics are most nonlinear, i.e.,30 to 40 degrees angle of attack.

Since it is most important that the simulation model predicts the actual in-flight dynamic response, both the current WL model and the fully nonlinear model were trimmed to the flight test condition and driven using the flight test stick time histories. In addition, care was taken to ensure that all FCS

functions were set according to the flight log. All weight and inertia information were calculated based on fuel state.

Although many test maneuvers were simulated, only one is presented here. A 35° α wind-axis roll maneuver (stick-driven) was selected because, like most aircraft, the F-16 exhibits the most nonlinear flight-dynamic characteristics near the stalled region. The flight is near trim at 35° α and descending slightly at 25,400 ft. The true airspeed is approximately 260 fps and the power is in military setting (this is the same flight used to generate the data in Figures 5 and 6). It should be noted that a significantly large amount of right stick was input by the pilot at the beginning of the maneuver to counter an apparent roll asymmetry which is included in the nonlinear database.

Figure 8 shows the response of the current WL F-16/MATV model to stick inputs identical to those of the flight data shown in the same Figure. The correlation is poor, showing lack of angle-of-attack hold (aircraft simulation pitches to over 60° α after 8 seconds) and a much lower wind-axis roll rate is exhibited by the simulation model. The inability to hold angle of attack is particularly detrimental in this case, due to an FCS mode that was set which inhibits rolling maneuvers at higher angles of attack. It should be noted that a 35° α wind-axis roll maneuver can be executed using the current WL model without pitching to higher α, however, the required stick inputs would be significantly different.

Figure 9 shows the same comparison using the fully non-linear model. The angle-of-attack hold compares well to that of flight test, as does the wind-axis roll rate. However, the sideslip, although showing similar characteristics at the beginning of the maneuver, becomes far too oscillatory. Since check cases evaluated in Reference 2 and the data presented herein had indicated that the static data are representative, modeling deficiencies in the dynamic data were suspected, specifically in the body-axis damping terms. This anticipated result was based on the revised low speed database's rigorous application of test data. The original linearized F-16 forced oscillation test data exhibited a wide range of non-repetitive values in the post stall region, and some judgment was required to define the baseline configuration from these data. Based on these and other considerations, the confidence level in this portion of the model was minimal. To illustrate the sensitivity of the post stall motion to these damping terms, the maneuver was repeated changing only the dynamic derivative which models body-axis rolling moment due to yaw rate, or C_{l_r}. The stability of this derivative were increased between 20° to 45° α, but left unchanged elsewhere. Figure 10 shows the 35° α wind axis roll with this change. Notice that this simple change positively affected the correlation of all three parameters presented, distinguishing the strong dependence of high-angle attack maneuver modeling on dynamic data.

Conclusions

The F-16 model currently used at Wright Laboratory has proven to be a highly effective tool for predicting flight response of the F-16 at low-to-moderate angles of attack. However, due to portions of the aerodynamic model that are represented by linearized data, enhancement of the database near stall and at high angles of attack and sideslip was required to increase simulation fidelity. This has been successfully demonstrated, but this study indicates that further investigation of dynamic effects, especially near stall, should be undertaken to further increase model fidelity.

In addition to showing how high angle of attack modeling of the current Wright Laboratory F-16/MATV model can be enhanced, this study has shown that today's personal computers provide the computational power necessary to accurately re-host the most complex simulation models of fighter aircraft being assembled. Re-hosting, evaluating and updating complex simulation models using PC-based tools such as those presented in this paper will significantly increase the efficiency of many aspects of aircraft development, from preliminary design to flight test, by transferring real-time, high-fidelity simulation from the expensive and shared-resource mainframes and workstations to the inexpensive and readily-available personal computer.

References

1. McKeehen, P., " F-16/VISTA High-Angle-of-Attack Aerodynamic Modeling and Analysis," WL-TM-91-304, February 1991.

2. Kay, J., Ralston, J., Lash S., "Development of a Non-Linear, Low-Speed Aerodynamic Database for the F-16/VISTA," AIAA-97-3576, August 1997.

3. Baxter, J. et. al., "Flight Test Demonstration Report for the F-16 Multi-Axis Thrust Vectoring Program," WL-TR-95-3086, January 1995.

4. Ralston, J., Kay J., "The Utilization of High Fidelity Simulation in the Support of High-Angle of Attack Flight Testing," AIAA-96-3354, July 1996.



Figure 1.- Variable-Stability In-Flight Simulator Test Aircraft with Multi-Axis Thrust Vectoring.

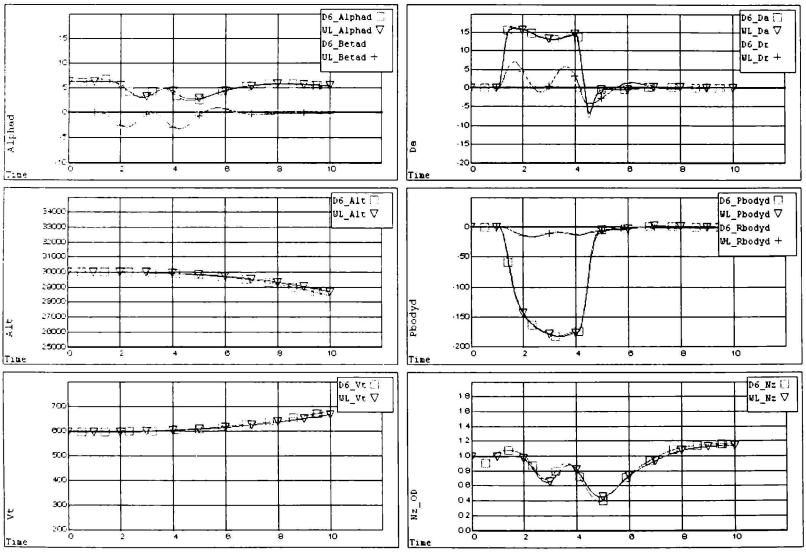
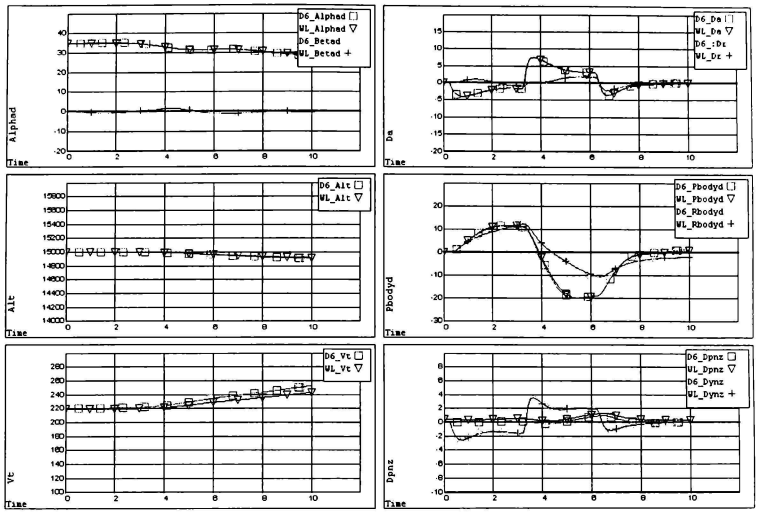


Figure 2.- Comparison of re-hosted model to WL model - F-16 Block 40 maximum roll command check case.



MATV Lim_Off: 35 alpha roll doublet

Figure 3.- Comparison of re-hosted model to WL model - F-16 with MATV on roll doublet check case.

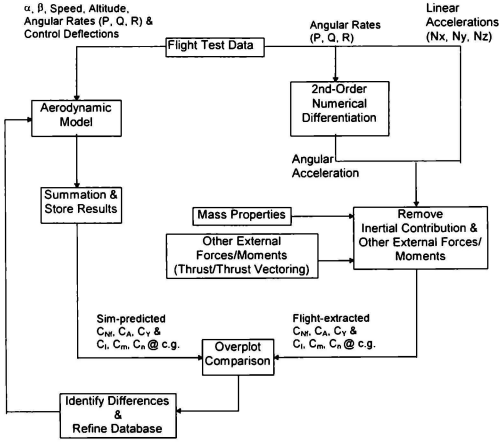


Figure 4.- Flow chart representation of the **Overdrive** feature of **D-SIX.**

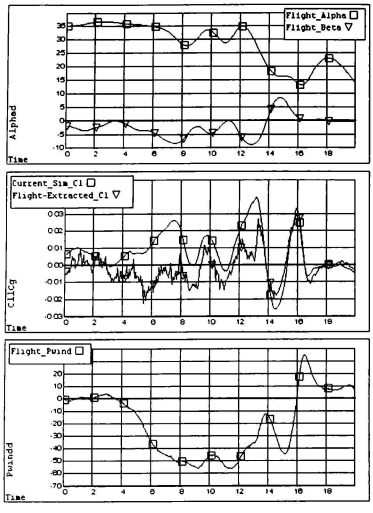


Figure 5.- Comparison of flight-extracted to simulation-predicted rolling moment coefficient using current WL database.

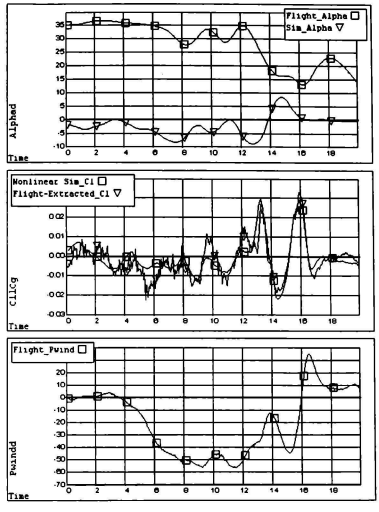


Figure 6.- Comparison of flight-extracted to simulation-predicted rolling moment coefficient using fully-nonlinear database.

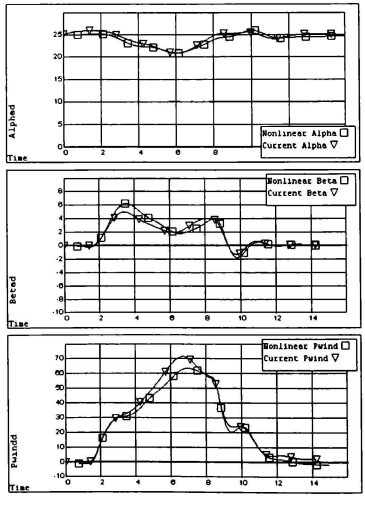


Figure 7.- Comparison of current WL model and fully-nonlinear models executing a limiter roll.

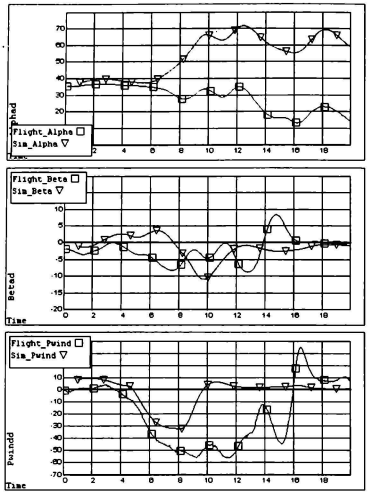


Figure 8.- Comparison flight test and simulation predicted 35° α rolling maneuver using current WL database.

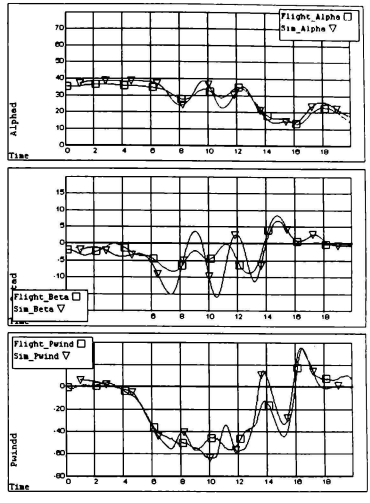


Figure 9.- Comparison flight test and simulation predicted 35° α rolling maneuver using nonlinear database.

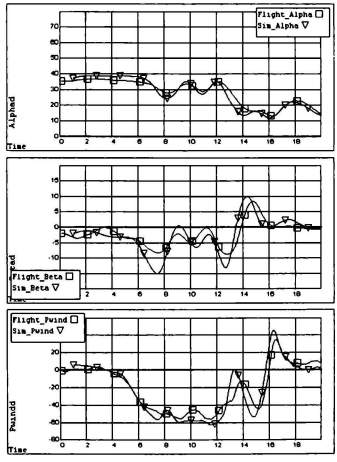


Figure 10.- Comparison flight test and simulation predicted 35° α rolling maneuver using nonlinear database with increased roll due to yaw near stall.

AIAA-97-3798

Application of a Multipurpose Simulation Design

Jeffery W. Bell*, Matthew J. O'Rourke'
Bihrle Applied Research, Inc.
Hampton, VA 23666

Summary

Seven years ago, Bihrle Applied Research (BAR) began developing high fidelity simulation software on PCs. At the time, the only driving factors were cost and availability. Some inexpensive PCs were available for simulation research, while the better-equipped and more expensive workstations were not. The PCs used in the early research phases were slow and had limited resources, and were generally far less capable than the workstations on which most high fidelity simulations were hosted. This paper discusses the problems BAR encountered developing a high fidelity simulation in a limited PC environment, the decisions that were made to resolve these problems, and the effect some advances in PC technology had on original design goals.

Introduction

Despite the limitations imposed by the PC environment, BAR set ambitious goals in the original design specifications with the assumption that PC hardware and software technology would continue to advance:

1. High fidelity simulation with fully model-independent operation capable of both rapid prototyping and full system simulation.
2. Modular structure allowing easy configuration without programming to support the diverse requirements of the aerospace industry.
3. Integrated simulation development and analysis tools.
4. Object oriented design supporting extensions without code changes to core components.
5. Flight simulator capable of fully modeling an aircraft in real-time on a PC running 32-bit Windows.
6. Reusable tools available to different applications associated with flight simulation, but not necessarily dependent on it.

The term model-independence indicates an application capable of operating with any or no simulation model present. In effect, the simulation model is read from one or more files, and can be changed without altering or restarting the application. As an independent contractor, BAR works with a number of different aircraft models and aerodynamic data sets, including data obtained from short wind tunnel tests targeting the behavior of a specific aircraft configuration in a specific region of flight. Separating common simulation functionality from aircraft-specific behavior was essential to allow aerospace engineers to concentrate on modeling aircraft without becoming software engineers in the process. Additionally, it allowed the simulation developers to concentrate on producing good software without becoming aerospace engineers in the process.

General Structure

One of the first problems encountered early in the design process was the need for including program code in the simulation model. Different types of aircraft with significant differences in flight characteristics, different database mechanization techniques selected by different engineers, the need for a flight control system, and evolving simulation technology all contribute to the complexity of the simulation model. These factors make it nearly impossible to develop a simulation model for use in a high fidelity simulation without writing some minimal amount of code.

The ideal solution would have been an interpreted language built into the simulation application. The simulation application could read all data and code associated with a given simulation model directly. Unfortunately, interpreted languages are, by definition, slower than compiled languages, and performance is always a concern in a PC environment. Additionally, porting simulation models between other simulation platforms would be more difficult if it involved converting between different programming languages. And while neither portability nor language independence was a specific design goal, they are nearly always goals in

* Software Engineer
' Aerospace Engineer

Copyright © 1997 by the American Institute of Aeronautics and Astronautics, Inc. All rights reserved.

a good software design as long as they do not compromise other more critical goals.

The Windows operating system (OS) provided a relatively simple solution using Dynamic Link Libraries (DLL). A DLL is a library of functions with built-in support for run-time linking. In effect, a DLL can be loaded into an application's address space after the application has been started. Functions exported by the DLL can be imported (called) by the application and functions exported by the application can be imported by the DLL. In reality, the only difference between an application and a DLL is an application begins and ends the process. Both application and DLL are referred to as program modules, and are programmatically associated with identifiers (handles) to distinguish them within a process.

Executable code required for the simulation model was therefore placed in a DLL, and loaded by the simulation application at the same time as the simulation model's data was loaded. Additionally, the majority of the code directly associated with building and running a simulation model was moved out of the main application to a second DLL called the simulation library. The main application was to provide the user interface and graphics, and other functionality not directly related to computationally simulating an aircraft. The model library provided a convenient location to place functionality used by various tools not directly related to running the simulation. Applications other than the simulation application could load the simulation library and immediately gain all the functionality needed to operate on a simulation model. Figure 1 shows the general structure of the planned simulation environment.

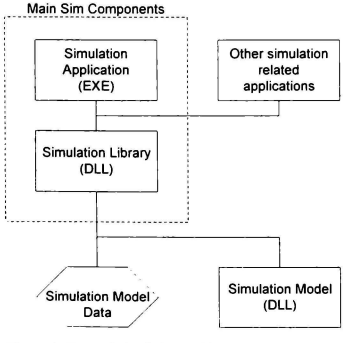


Figure 1. General simulation architecture

Moving the simulation model program code out of the simulation application and into a DLL required a means of sharing data between the two modules. With the performance and resource constraints of an application running on an early Windows based PC, it was not possible to pass data as function parameters. The overhead of loading and unloading the stack had a dramatic effect on performance on the 486 33MHz machines used in the original design implementation.

A common database was used instead, that was shared by all modules within the running simulation. Since simulation application, library, and model DLL all share a common address space within the OS, the problem became more logistical than technical. Data and function pointers are commonly passed between modules in Windows applications, such that it is relatively simple to allow every module within a given address space to access a given variable. However, knowledge of the variable must exist at compile time. The simulation library was given control of a simple heap structure that exported pointers to key elements within the heap that would allow different modules access to the database. For example, a simulation model's DLL could access a variable "Alpha" with the following line of C-code:

```
pSimData [ 0 ] [ GetIndexToName ( "Alpha" ) ] = MyAlpha;
```

Note, the first array index "[0]" is required because Windows can only export pointers to data. pSimData is an array in the simulation library, so it is exported as a pointer to an array. The first pointer resolution is required in the simulation model DLL to convert pSimData back into a simple array.

While this solved the logistics, it had an undesirable impact on simulation performance. Each variable accessed within a module other than the simulation library would be required to indirectly address an array through a string search function. Performance was increased marginally by looking up the required names prior to performing a simulation run, and caching the offsets in a different array. The resulting lines would appear as follows:

```
// Before performing a simulation
iSimData [ ALPHA_INDEX ] = GetIndexToName ( "Alpha" );

// During a simulation
pSimData [ 0 ] [ iSimData[ ALPHA_INDEX ] ] = MyAlpha;
```

On Intel 486 and Pentium processors, the effects of this indexing scheme were somewhat surprising. The processor maintains a single pass indexing capability for

up to four levels of indirection. In effect, the processor should be capable of reading the desired value in the above line in a single read operation as long as there are no more than four different offsets associated with the read operation. However, the line of code actually has 5 levels of indirection: *PSimData, [0], iSimData, [ALPHA_INDEX], and [iSimData [ALPHA_INDEX]].* The resulting machine code two read passes, and causes two separate cache flushes in this single line of code. As stated above, the performance increase over a text-based index search was only marginal.

By assigning the value of *[iSimData[ALPHA_INDEX]]* at compile time, the level of indirection in the previous line could be reduced to three. To implement this, a preprocessor was implemented in the simulation application to generate source code definitions for each shared variable. The above lines could then be converted to the following:

```
// source definitions generated by the simulation application
#define Alpha pSimData [ 0 ] [ ALPHA_INDEX ];

Alpha = MyAlpha;
```

This approach allows the database to be defined completely within the running simulation library by the user and preprocessed to create variable definitions in the simulation model's DLL. The simulation model's DLL is then compiled and loaded into the running simulation application. The entire process requires less than a minute in most cases, and is only required when there is an addition or deletion of a variable in the database that is needed by the simulation model's DLL. The simulation performance of this method approaches that of the traditional combined simulation application and model.

While sharing a common database between the simulation application and the simulation model solves a number of performance related problems, it is a significant deviation from traditional object oriented programming (OOP) techniques in that it provides no encapsulation. The simulation model developer is free to directly access data inside the heap object created within the simulation application. However, the preprocessing capabilities built into the simulation application provide a simple approach to accessing variables in the database that is consistent with OOP methods. Simulation model developers are also discouraged from accessing the database directly. In any case, the performance gains obtained from sharing the database in this manner were considered significant enough to justify this deviation from OOP.

While the running application provides the interface to add and delete new variables, for performance reasons it is restricted to times when the simulation is not running. An interface defined as a heap manager is exported by the simulation application to support database manipulation in a more object-oriented manner than when the simulation is running. In effect, there are functions available for adding and deleting variables, performing searches, and classifying data types. Encapsulation is performed at the interface level rather than in the language. All modules are required to manipulate the database through the provided interface, since the exported database variables only point to the current heap during a simulation run, and performance is not a critical issue while the simulation is not running.

The simulation model was separated into two basic components. The first was the executable functions built into the model's DLL, and the second was the database used by both the application and model code. Additional data sources included smaller blocks of data, such as initial conditions, control and stick deflections, etc. The original design specified a different file, and extension for each data type, such that each was stored and loaded as a separate entity. However, preliminary testing showed engineers found the overhead required to get the simulation up and running to be somewhat oppressive. To simplify simulation operation, all of the files were stored in or referenced by a single *project* file. The database, initial conditions, and table look-up information were placed directly in the project file, while the model's DLL and the tabular table data were referenced by file name to reduce the size of the project file. Support for each of the separate file types was also retained to allow models to be assembled from different pieces, so different individuals could work on different portions of a model simultaneously. Figure 2 shows the different components of a typical simulation model.

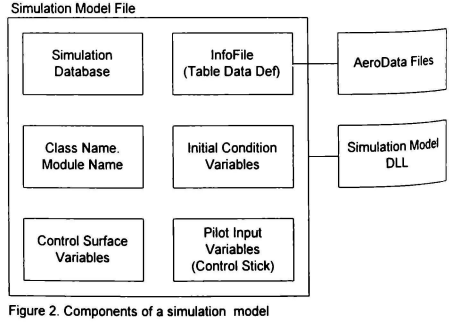


Figure 2. Components of a simulation model

Aerodynamic Database

Probably the most difficult problem encountered was developing a data table structure that supported both rapid prototyping and full simulation model development, and did not prevent the simulation from operating in real-time. Preliminary design goals required support for tables of any dimension that could be resized and edited without source code changes. To support in-house editing tools, and to improve source control operations, the actual data was to be stored as a number of two-dimensional data files that included indexes for both dimensions and an index value for an optional third dimension.

BAR had already adopted a two-dimensional data file format as the basic building block for all data tables of any dimension. A number of in-house tools had been developed to manipulate the data in the files, and provided useful functionality that was not planned for the first version of the simulation application. The existing format was therefore included with only minor changes as the supported internal data table format (BAR 2.0). The design included a second definition file (InfoFile) to define multiple dimensional tables as groups of two-dimensional data files. The InfoFile contained the names and order of the data files used in each look-up operation, as well as table dimensions, independent variables and other information not contained in the BAR 2.0 files. Since assembling the table data was considered one of the more complex tasks of developing a simulation model, a significant effort was expended to produce a simplified visual interface for the task. Figure 3 shows the basic structure of a typical look-up table in the InfoFile.

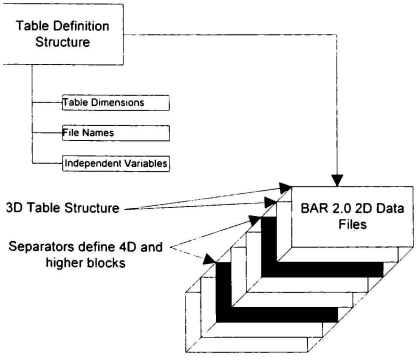


Figure 3. InfoFile Table Definition

Since the InfoFile was designed to support the complex editing features of a visual interface, table look-up operations were predictably too slow to support real-time simulation. A second data table format (DTF) was developed in parallel to the InfoFile to provide the actual table look-up functionality during a simulation. A DTF file is essentially a compiled form of the InfoFile and related BAR 2.0 data files. A number of performance optimizations were implemented, such as redundant index and table removal, last value comparisons, and index boundary crossing detection. Since the DTF file data could always be regenerated from the InfoFile, it could be optimized without regard to user access, data editing, or table identification. A highly optimized table look-up engine was developed to work with the DTF file data that required an average of 75 nanoseconds to perform a table look-up on a single dimension of a typical data file. Some additional optimization was achieved by restricting tables to a maximum of sixteen dimensions, and a small section of the table look-up code was converted to assembly language when the C++ compiler failed to produce the expected optimizations.

Simulation Model Extension Interface

Flight simulation technology is still an evolving technology. A significant amount of research is still being performed to better model different aircraft, and new techniques are emerging constantly. Because of this, the simulation application was designed to allow simulation models to replace any simulation functionality normally provided by the application. In effect, the simulation model could replace any function contained within the simulation application that was called during a simulation. This was to allow simulation technology researchers to test their theories using a stable, well-tested simulation application, and to compare their results with existing methods. The researchers could then concentrate on only the portions of the simulation their research involved without developing a new application, and with insurance the rest of the application was running as expected.

To support the replacement of simulation functionality within a simulation model, the simulation application would search the list of exported functions within the simulation model's DLL for a name matching a specific set of replacable functions. If a replacable function was found, the application called the function provided by the simulation model rather than its own. The researcher had only to provide a function with the correct name, and the source code for the replacable functions were made readily available as a starting point to further

reduce the learning curve. A simple list of function pointers was used to reference the correct functions. While there was a significant increase in function call overhead compared to calling a function directly, the number of required indexed function calls was too small to measurably affect simulation performance.

External Development and Control Interfaces

The addition of extensibility to the simulation application was not restricted to simulation technology research. A real-time high fidelity simulation has a number of applications in other fields, such as training and interactive war gaming, etc. The simulation design included several interface specifications to load specialized or generic modules to implement otherwise unsupported functionality. An I/O interface was included to provide a relatively simple method of connecting the simulation to various instruments, switches, and control devices. A graphics interface was included to support external displays, intercoms, and sound devices. Finally, a generic interface was included to support unknown functionality. The graphics and I/O interfaces are derivations of the generic interface. The primary difference being optimizations built-in to the simulation application to support the non-generic implementations with less overhead.

The generic interface supports starting and stopping the simulation, loading and saving simulation models, and menu additions. It was originally intended as a supporting interface for networking and control software. However, several tools have been developed and incorporated into the simulation application using the generic interface. For example, a control system design tool capable of reading block diagram specifications and converting them to source code was implemented in this manner. Another tool is currently being developed to automate the process of editing and compiling a simulation model's source code.

Networking modules have also been implemented using the generic interface. However, an extension to the interface, the Module Communication Interface (MCI), was required to manage data definitions, conversions, and message routing. MCI provides network aware modules with registration and message services, and maintains a separate, reentrant database interface for storing time-dependent data values for different objects participating in a simulation. MCI does not provide network services, define networking protocols, or require specific data types or formats. Instead, it provides a common interface suitable for binding the communication and data definitions used by most networking services. For example, an existing proprietary network interface developed specifically for verifying the functionality of the generic interface and MCI is not directly compatible with the Distributive Interactive Simulation (DIS) standard. However, a DIS compatible MCI module can communicate with objects communicating over the proprietary networking interface through MCI. In effect, MCI is a transport layer designed to bind different network protocols together, and is optimized for use with interactive simulation applications.

Component Object Models

All of the interfaces designed for the simulation application were implemented as a list of functions exported by the module implementing the interface, and imported by the modules connecting to the interface. While this approach offers encapsulation, with the previously discussed exceptions, it lacks true polymorphism. The behavior of an interface function can be changed by replacing the DLL that implements it, but that would mean altering known behavior of the interface. A more suitable OOP approach for defining behavior between modules is with a component object model (COM) specification.

While Microsoft has supported COM through a proprietary interface specification called object linking and embedding (OLE) since the release of Windows 3.1, BAR has not adopted it due to the number of alternative specifications used on different (non-PC) platforms, the incompatibilities between them, and the fear the Microsoft would discontinue or significantly alter OLE/COM. However, recently several UNIX operating systems on PC's and on workstation computers have agreed to support the OLE/COM interface, and Microsoft plans to convert the traditional Windows GDI into a COM interface in the next release of Windows NT.

BAR plans to provide COM interfaces as alternatives to all existing interfaces for version 2.0 of the simulation application (D-SIX) scheduled for release in January 1998. This will provide a means of distributing the functionality of a simulation across multiple computers without a proprietary interface specification. Additionally, the interface will be platform independent, allowing transparent interoperability with other computer hardware not running Windows. For example, the program code for a simulation model could be implemented on a different computer with a different operating system than the computer running the simulation. The interface specifications will be made freely available to any interested parties.

Performance

Throughout the development of the simulation software, the initial design has remained surprisingly constant. The restrictions imposed by the limitations of the PC/Windows environment required a number of compromises internally to address performance problems. However, the interfaces between each program module remain almost identical to those originally designed. The most significant impact of the PC/Windows environment was the amount of time required to optimize the source code, since the simulation design was originally intended to operate in real time on 486 machines.

With the performance of PC's today surpassing the performance of workstations used for simulations less than three years ago, source code optimization has become more of a habit than a requirement. However, a high fidelity simulation has applications in training and interactive simulations that require additional processor time. The ability of the simulation to function in real-time while running add-on modules for high-resolution graphics and interactive networking can be attributed primarily to the rigorous optimizations implemented throughout the development process.

The current implementation of the simulation application (D-SIX) is capable of running extremely complex simulation models in real-time on most standard PC's. For example, the F18E/F model imported into D-SIX from the MDA model contains more than eight hundred multidimensional look-up tables and almost four thousand database variables. The simulation runs the F18E/F model in real-time at 80Hz on a 166MHz Pentium machine. This includes a port of the full flight control system consisting of more than twenty thousand lines of Fortran code.

User Interface

One of the advantages of developing software on an inexpensive PC is the relatively low cost of advanced development tools. The large number of software developers producing software for the PC has allowed compiler manufacturers to expend enormous resources on producing advanced development tools and still distribute them at relatively low prices. Additionally, since the majority of the applications for the PC are for non-technical fields, such as business applications and games, development tool advancement has been largely in the direction of improving user interfaces and simplifying usage through graphical or *visual* interfaces. The development of D-SIX profited significantly from these tools. Simulation model development and analysis complexity was reduced considerably compared to more traditional simulation applications.

The display interfaces shown in figure 5 illustrates a few of the graphical user interfaces used in the simulation environment. These interfaces control operations ranging from simulation independent variable definition and input, model initial definition and setup, control surface time-history editing, as well as graphical editing of any table-based terms. It is the extensive use of graphical interfaces such as these that allows the engineer to conduct virtually all of the typical simulation development, analysis, validation, and deployment operations with little or no code level interaction.

Conclusion

Performance issues related to PC limitations were a significant consideration on the original software design. While current PC performance has increased by orders of magnitude from the original design platform, coding efficiencies have yielded the current capability to host the most complex engineering simulations in real-time on a PC.

The idea of the original design was to produce a simulation platform that could be easily expanded as hardware and operating system technology advanced. While new technology and better PC performance has significantly enhanced the capabilities of the software, the original interface specification has changed very little. The flexible interface design has allowed the rapid development of new capabilities, such as high-resolution graphics, and interactive networking. New capabilities and new simulation technology can be added without changes to existing software through add-on modules and simulation model extentions. As long as developers produce software compatible with the existing interface design, compatibility of simulation models, add-on modules, and the simulation application is guaranteed.

References

Petzold, C., *Programming Windows: The Microsoft Guide to Writing Applications*, Microsoft Press, Redmond, WA, 1992. ISBN 1-55615-395-3.

Brockschmidt, K., *Inside OLE*, Microsoft Press, Redmond, WA, 1995. ISBN 1-55615-843-2.

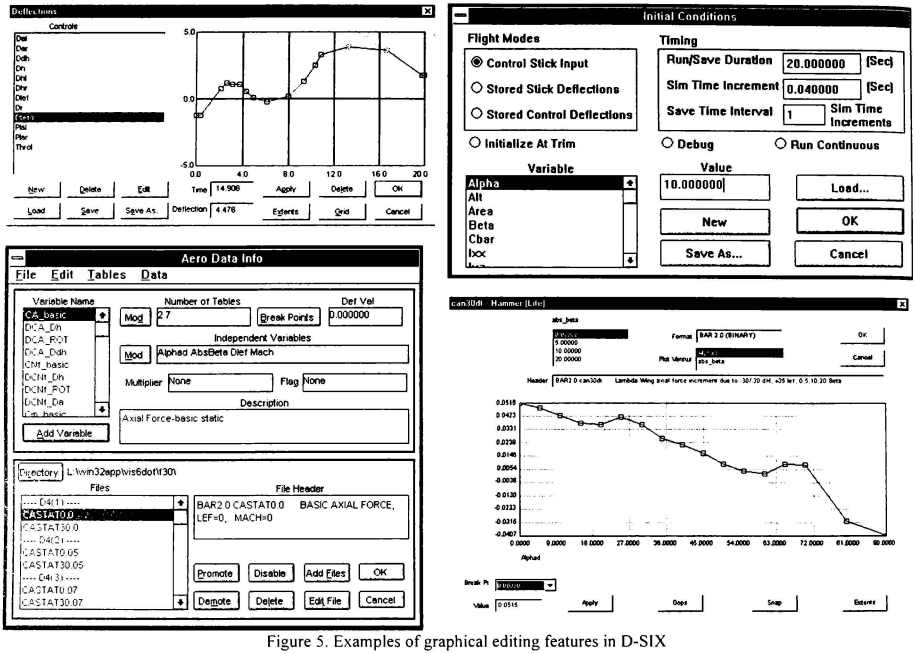


Figure 5. Examples of graphical editing features in D-SIX

AIAA-97-3523

THE TISES MANY-ON-MANY ENGAGEMENT PLANNING ALGORITHM

James R. Ashburn (Nichols Research Corporation, Huntsville, Alabama 35802)
Dawn Horn (THAAD Project Office, Huntsville, Alabama)

A missile defense many-on-many engagement planning algorithm designed for the THAAD Integrated System Effectiveness Simulation (TISES) is described. The algorithm represents an effort to fulfill as many of the features envisioned by the THAAD community for the objective system as possible. This paper describes the basis for the algorithm's objective functions, the constraints imposed on the problem, and the various defense objectives that may be represented. It addresses issues such as inventory-limited defense, overlapping asset damage, and drawdown strategies. The paper concludes with discussions of "adaptive method-of-fire," the strengths and weaknesses of predictive ("look-ahead") engagement planning, software implementation issues, and an integrated approach to engagement generation and selection.

DISTRIBUTION STATEMENT A. Approved for public release; distribution is unlimited.

Introduction

The many-on-many engagement planning problem is among the most well known battle management problems facing missile defense systems. Engagement planning software must seek to fulfill defense objectives within a large set of constraints while minimizing resource utilization. To the extent the driving factors can be regarded as random, engagement planning is a stochastic control problem, not unlike that of an interceptor nulling a predicted miss distance.[1,2] In the case of engagement planning, the parameters to be controlled include the final (i.e., foreseeable end-of-battle) states of the threats, defended assets, and interceptor inventories. As a control problem, the battle manager must periodically reevaluate the perceived state of the threat and the current response to that threat and determine the best changes to the response to achieve the desired balance between fulfilling defense objectives and conserving resources for detecting and negating future threat objects.

The algorithm described herein was developed for TISES, a high-fidelity, discrete-event, end-to-end missile defense system simulation, to address the capabilities envisioned for the THAAD objective system.

Among the capabilities envisioned for the objective system was the probabilistic treatment of projected interceptor inventories. This simple application of the mean-value approximation is often referred to as "fractional rounds" by much of the THAAD BM/C^{4}I community and was a major driver in the selection of an approach to engagement planning in TISES. Because the idea of probabilistic treatment of resources had already been documented by AlphaTech in a published description of their Look-Ahead Battle Planner (LABP),[1] predictive (or "look-ahead") planning was therefore adopted as a pattern for the TISES algorithm.

Purpose

Whereas the purpose of the LABP was to "project how the situation will look at some future time if the defense executes a specified CoA [Course of Action]," the algorithm described herein seeks to use these projections as the basis for an objective function that can simplify or augment preplanned rules, priorities, and constraints, such as those in the NMD CoAs or the TMD Engagement Subplans. The original goal was to develop an algorithm that required that the operator of the system specify only the performance objective while the algorithm determined the response that best achieved it within the limitations of the system. All other inputs to the algorithm were presumed to represent measurable physical capabilities of the system.

References

[1]The Look-Ahead Battle Planner, John J. Shaw, Mark A. Gerber, Ronald M. James, Major Anthony P. Rizzuto, Proceedings of the 1993 Summer Computer Simulation Conference, sponsored by The Society for Computer Simulation, P. O Box 17900, San Diego, CA 92177, pp. 717-726.

[2]To the extent the external stimuli are not random or at least not so regarded, such as the actions of an enemy in coordinating an attack, the engagement planning problem is a game theory problem. The implications of this fact appear to be well beyond the scope of the engagement planning art as it exists today. Personal communication, Gary Makowski, TecMasters, Incorporated, 30 May 1996.

The focus of this paper is to describe the basis for the TISES many-on-many algorithm and to highlight its relative merits, and identify its potential weaknesses. The rigors of the mathematicl basis for the algorithms are summarized to comply with security restrictions. Only a brief discussion of software implementation issues will be provided. No sample results will be presented is this exposition.

Terminology

Some discussion of the terminology used in this paper is beneficial at this point. First, the terms "engagement" and "shot" are often used interchangeably in reference to the utilization of a single interceptor against a threat object. Depending upon the context, these terms may also include the expenditure of radar resources necessary to support the interceptor in flight.

The terms "non-interactive" and "interactive" constraints must also be defined. Non-interactive constraints are timing and geometry restrictions that are imposed on an engagement independent of all other engagements. Examples include kinematic accessibility, sun/moon/earth limb exclusion angles, and time of flight limits. Interactive constraints, as the name suggests, involve dependent interactions between engagements usually associated with timing and resource limitations. Examples include interceptor inventory at a launcher and maximum launch rates.

Next, the terms "perceived" and "projected" situations must be clarified. The perceived situation includes the current perception of the states of the objects such as assets, threat objects, and defense resources, including those currently being employed or available for employment against the threat. Associated with each perceived state of an object is a projected state corresponding to an estimate of its condition at the foreseeable end of the battle.

The terms "existing," "candidate," and "planned" engagements must also be defined. Existing engagements are those which are currently being executed. To be more precise, they include all engagements currently committed against a threat for which kill assessment has not been completed. Candidate engagements represent feasible potential engagements against a given threat; they are "candidates" for execution. Each is feasible in the sense that it can be executed if no other candidate engagements are executed. To qualify as a candidate, the engagement must first fulfill all non-interactive constraints. Second, the candidate engagement must not violate any interactive constraints associated with any existing engagements. Planned engagements are those candidate engagements that are expected to be subsequently executed contingent upon the perception that the associated threat is still alive at the required commit time. Only the planned engagements that require immediate commit are executed in a given planning cycle; the remainder are used only for estimating the outcome of the battle.

Next, the terms "execution," "kill quality hit," and "kill" must be clarified. The terms "execution" and "commit decision" are effectively synonymous. The use of the term "execution" is not meant to imply successful completion of an engagement, only a commit decision and an attempt to perform an engagement. The term "kill quality hit" is used to distinguish a hit of sufficient nature to have killed a threat (independent of the status of the threat before the hit) from a kill (a hit that renders a live threat inactive).

The Mean-Value Approximation

Projecting the possible outcomes of a battle involves combinations and permutations of many conditional events. For a problem of even modest size, enumeration or even sampling of the possible outcomes is impractical. Consequently, the mean-value approximation is employed, hereinafter referred to as MVA. MVA replaces enumerations of possible outcomes with the associated expected (or mean) outcome. Shaw, *et al.* provide an excellent discussion of this concept.

Some examples of the implications of MVA are worthy of mention. First, perceived states that are measured as discrete quantities, such as interceptor inventory, have continuous projected state counterparts. For example, consider a launcher with ten interceptors and a planned shoot-look-shoot method of fire against a threat, each with an expected 90% chance of success. The projected remaining inventory is then 8.9 since the probability the second interceptor will be used is only 10%. Second, no distinction is made between binary discrete states (e.g., operational/not operational) and their continuous counterparts (e.g., X% operational). For example, one threat that has a 90% chance of destroying an asset and a 10% chance of doing no damage to the asset is indistinguishable from a second threat that will always destroy exactly 90% of the asset.

The Simplified Approach

To formulate an initial solution to the engagement planning problem, this author has attempted to decompose the problem into the greatest number of serial processes without sacrificing the quality of the solution. In other words, a decision to integrate any two processes is based upon its necessity to retaining the quality of the solution; runtime and memory are not priorities in this *simplified approach*. These considerations are discussed in a subsequent section on

the integrated approach. The processes that have been identified are as follows:

- Situation Maintenance
- Threat Assessment
- Candidate Engagement Generation
- Candidate Engagement Selection
- Engagement Task Message Generation

The first process, Situation Maintenance, is required to insure the most up-to-date representation of the situation as it is perceived by the battle manager. The process operates upon the perceived situation data and notes state transitions that occur as a result of the passage of time. Namely, these consist of assumed state transitions as a result of overdue messages (i.e., incoming data, either periodic or scheduled, which has failed to arrive at its expected time). These include the determination of stale tracks (e.g., overdue periodic track report), presumed non-operational defense resources (e.g., overdue periodic status report), and presumed failed tasks (e.g., overdue scheduled reports such as interceptor status reports and kill assessment reports). Many of the messages incoming to a battle manager fall into this category of "anticipated" messages.

The second process, Threat Assessment, determines which threat objects are eligible for engagement. If a threat value-based objective is selected, threat objects are assigned values consistent with their damage potential. In the case of an asset value-based defense objective, threat objects must be associated with the assets they are likely to damage. A mean damage estimate for each threat/asset combination is also required.

The third process, Candidate Engagement Generation, calculates a set of feasible engagements for all threat/launcher/supporting radar combinations to some sufficient time granularity. This process can be equated to identification of the solution space for the optimization problem that is to be solved in the next process. A good rule of thumb is to generate at least one engagement commit opportunity per planning cycle within non-interactive constraints for each threat/launcher/radar combination. First, this insures that at least one engagement opportunity for each threat/launcher/radar combination is available for immediate commit if feasible so that engagement opportunities are not deferred indefinitely. Second, it insures reasonable temporal packing for shoot-look-shoot and salvo engagements. Alleviating the computational burden of generating a potentially large number of engagements is discussed in a later section.

The fourth process, Candidate Engagement Selection, is the focus of this paper; it is the process which contains the many-on-many algorithm. This process must select from the set of candidate engagements a set of planned engagements expected to best fulfill the defense objectives. These planned engagements include both those that may require an immediate commit decision along with those whose commit decision must be deferred. The latter include those whose commit decision may be conditioned upon the outcome of other engagements.

Separation of the third and fourth processes into serial operations is the identifying characteristic of *the simplified approach*. In a subsequent section, an integrated approach to these processes is introduced.

The fifth and final process, Generate Engagement Task Messages, selects from the set of planned engagements those requiring an immediate commit decision and generates and transmits the messages necessary to initiate them.

Selection of an Optimization Method

Selection of an appropriate optimization method was based upon the following reasoning. Because independent probabilities are never additive, linear methods are dismissed immediately. Because of the complex nature of the constraints imposed upon feasible solutions to the many-on-many problem, all non-general optimization methods are precluded. Because of the combinatorial size of the problem and the speed with which it must execute, only the most rudimentary general optimization method remains viable, namely maximum marginal return (hereinafter referred to as MMR). Finally, because of the convoluted nature of the objective function and the constraints, only a suboptimal solution can be expected.*

Admittedly, MMR has some potentially serious deficiencies when faced with problems such as estimated kill probabilities that increase with the advancement of time and overlapping asset damage. However, these problems are not insurmountable and proposed solutions are discussed in later sections.

The Inputs and Outputs

The basic TISES many-on-many algorithm operates upon assets, threats, engagements, launchers, and launcher sites. These objects are identified by the following indices:

i = asset
j = threat
k = engagement
l = launcher
m = launcher site

* MMR is an optimal method only in the case of an objective function with no local optima.

Examples of associations between these objects are as follows:

J_k = threat j associated with engagement k
L_k = launcher l associated with engagement k
$\{K_l\}$ = set of engagements associated with launcher l
$\{K_{E,j}\}$ = set of existing engagements associated with threat j
$\{L_m\}$ = set of launcher s associated with site m

Note the shorthand notation used for set intersections. For example,

$$\{K_{E,j}\} = \{K_E\} \cap \{K_j\}$$

In the following, inputs and outputs associated strictly with leakage, threat-value, or defense-value defense criteria are identified by (L), (T), and (A), respectively. Summary outputs include the following:

A = projected total surviving asset value (A)
A' = linearized projected total surviving asset value (A)
B = projected total remaining interceptor inventory
B' = projected total remaining interceptor value
N = projected total number of threats negated (L)
H = projected total negated threat value (T)

The inputs for each asset i are as follows:

A_i = current value of asset i (A)

The corresponding outputs are

v_i = projected surviving fraction of asset i (A)
v'_i = linearized projected surviving fraction of asset i (A)

The inputs for each threat j are the following:

F_j = nominal damage capability of threat j (T)
$F_{i,i}$ = mean fraction of asset i destroyed by threat j if not engaged (A)
$\{K_{E,j}\}$ = set of existing engagements against threat j
$\{K_{C,j}\}$ = set of candidate engagements against threat j

Convention would normally dictate a set of decision variables associated with the candidate engagements. In lieu of such decision variables, an initially empty set of planned engagements is introduced. Equivalent to toggling a decision variable, a candidate engagement is moved to the set of planned engagements. Consequently, the threat outputs are as follows:

P_j = projected probability of leakage of threat j
$\{K_{P,j}\}$ = set of planned engagements against threat j

The inputs for each engagement k consist of

L_k = launcher associated with engagement k
$p_{QIE,k}$ = probability of kill quality hit given execution of engagement k (i.e., the single shot kill probability $p_{SSKP,k}$)
t_k = commit time of engagement k
τ_k = launch time of engagement k
T_k = intercept time of engagement k

In the case of existing engagements, p_{SSKP} is ideally the current estimate, appropriately adjusted to account for engagement performance up to the current time. For example, it may be increased for successful booster separation or decreased for low remaining divert fuel. The outputs for each engagement are

$p_{E,k}$ = probability of execution of engagement k
$p_{Q,k}$ = probability of kill quality hit of engagement k
$p_{KIQ,k}$ = probability of kill given kill quality hit of engagement k
$p_{K,k}$ = probability of kill of engagement k

The inputs for each launcher l are the following:

W_l = current interceptor inventory at launcher l
M_l = launcher site m associated with launcher l

Note that existing engagements have already been considered in determination of the current launcher inventories; the current inventory equates to the currently remaining available inventory. The launcher outputs are as follows:

w_l = projected remaining inventory of launcher l
w'_l = projected remaining interceptor value at launcher l

The outputs for each launcher site m are

w_m = projected remaining interceptor inventory at launcher site m
w'_m = projected remaining interceptor value at launcher site m

Inputs which describe system timing include the following:

t_c = current time
D = nominal shoot-look-shoot delay
$\Delta\tau_{min}$ = minimum time between launches at a launcher

The nominal shoot-look-shoot delay is the delay between kill assessment and subsequent commit for planned shoot-look-shoot engagements.

Rules of engagement inputs include the following:

R_{min} = minimum rate of return

ΔI = launcher inflation constant
ΔT = rate of return decay time constant
ΔD_m = deflation constant for launcher site m

The Model

For each engagement, three events are defined with which probabilities are associated— execution, kill quality hit, and kill, as defined earlier. The relationships between the engagement probabilities and their conditional counterparts can be written as,

$$p_{Q,k} = p_{QIE,k}\, p_{E,k} \tag{1}$$

$$p_{K,k} = p_{KIQ,k}\, p_{Q,k} \tag{2}$$

Assuming perfect kill assessment, the probability that a planned engagement will be executed is equal to the probability that the threat is still alive at the commit time for that engagement. For planned engagements requiring an immediate commit decision and existing engagements, the probability of execution must be unity. These conditions can be written as follows:

$$p_{E,k} = \begin{cases} 1 - \sum_{\substack{k' \in \{K_{E,j}\} \cup \{K_{P,j}\},\ j=J_k \\ T_{k'} < t_k - D}} p_{K,k'} & \text{if } k \in \{K_P\} \text{ and immediate commit not required} \\ 1 & \text{if } k \in \{K_P\} \text{ and immediate commit required} \\ 1 & \text{if } k \in \{K_E\} \end{cases} \tag{3}$$

If single shot kill probabilities (SSKPs) are assumed to be uncorrelated, then the probability of execution can alternatively be written as follows:

$$p_{E,k} = \begin{cases} 1 - \prod_{\substack{k' \in \{K_{E,j}\} \cup \{K_{P,j}\},\ j=J_k \\ T_{k'} < t_k - D}} \left(1 - p_{QIE,k'}\right) & \text{if } k \in \{K_P\} \text{ and immediate commit not required} \\ 1 & \text{if } k \in \{K_P\} \text{ and immediate commit required} \\ 1 & \text{if } k \in \{K_E\} \end{cases} \tag{4}$$

To avoid calculations of probabilities outside the range [0.0, 1.0] due to round-off errors, the form in the latter equation is preferred.

The probability of kill quality hit given execution is simply the SSKP, again neglecting correlations,

$$p_{QIE,k} = p_{SSKP,k} \tag{5}$$

The probability of kill given kill quality hit is simply the probability that the threat is alive at intercept given that it was alive at commit. This can be written as

$$p_{KIQ,k} = 1 - \sum_{\substack{k' \in \{K_{E,j}\} \cup \{K_{P,j}\},\ j=J_k \\ t_k - D \le T_{k'} < T_k}} p_{K,k'} \tag{6}$$

Again, assuming uncorrelated SSKPs, this can also be written as follows:

$$p_{KIQ,k} = 1 - \prod_{\substack{k' \in \{K_{E,j}\} \cup \{K_{P,j}\},\ j=J_k \\ t_k - D \le T_{k'} < T_k}} \left(1 - p_{QIE,k'}\right) \tag{7}$$

Given the equations above, estimates of the projected launcher inventories can be determined. These estimates are denoted by the following:

w_l = projected remaining inventory of launcher l

Utilizing MVA, the projected remaining interceptor inventory for a launcher is simply

$$w_l = W_l - \sum_{k \in \{K_{P,l}\}} p_{E,k} \tag{8}$$

The projected total remaining inventory,

B = projected total remaining interceptor inventory

is simply

$$B = \sum_{l} w_l \tag{9}$$

Interactive Constraints

Before addressing defense objectives and the objective function, a discussion of interactive constraints is fitting. The most prominent of these constraints is the interceptor inventory at each launcher. Consistent with MVA, the projected remaining inventory must be constrained as follows:

$$w_l \geq 0 \text{ for each launcher } l \tag{10}$$

Note that MVA does not guarantee that adequate inventory will be available for all planned shots. It is, however, an approximation that tends to become increasingly accurate in proportion to the size of the problem as it grows.

In addition to inventory constraints, timing constraints may also be present. Consider the example of a launch rate constraint at each launcher. Again, using MVA, this may be represented as follows:

$$|\tau_k - \tau_{k'}| > (p_{E,k} + p_{E,k'})\Delta\tau_{min} \text{ if } L_k = L_{k'} \text{ and } J_k \neq J_{k'} \tag{11}$$

$$|\tau_k - \tau_{k'}| > \Delta\tau_{min} \text{ if } L_k = L_{k'} \text{ and } J_k = J_{k'} \tag{12}$$

where

$\Delta\tau_{min}$ = minimum time between launches at a launcher

The first of these two inequalities approximates engagements on different threats as uncorrelated. The last inequality assumes that engagements on the same threat sufficiently closely spaced to violate a launch rate constraint have perfectly correlated probabilities of execution: that is, commit decisions are based on exactly the same kill assessment data.

Another constraint worthy of mention is radar loading. While the current implementation of the TISES many-on-many algorithm does not address radar loading, predictive planning in general provides estimates of the probabilities that radar support will be required. Because these probabilities may be quite low for follow-on shots, failure to adjust expected loading accordingly can easily negate any advantages of considering radar loading.

Defense Objectives

Given the model of engagement probabilities above, objective functions consistent with a number of defense objectives can be formulated. In this section, three are considered: minimize leakage, maximize threat value negated, and maximize asset value saved.

For a leakage-based defense, the goal is to minimize total leakage for any given expenditure of interceptors. The objective function to be optimized in this case is a trade between the number of threats negated and the number of interceptors remaining. To maximize the objective function for a given expenditure of interceptors in the context of MMR, one must select (one at a time) from the set of candidates the engagement that maximizes the objective function quantity. The process continues until a minimum rate of return can no longer be achieved. Note that *the minimum rate of return describes the aggressiveness of the defense response*; it defines when the expected marginal decrease in leakage is of more value than that of the expended interceptor. If set properly, it will insure that not all of the interceptors are expended on the first threats to be detected and become eligible for engagement.

As an alternative to a leakage-based defense, both threat value-based and asset-based defenses can similarly be represented.

In regard to the latter two strategies, it is important to recognize thatsince threats may damage multiple assets and assets may be damaged by multiple threats, no general simplified form exists. In addition, note that in each of the three defense criteria cases, the minimum rate of return parameter must be adjusted appropriately. In the case of leakage-based defense, the units are threats per interceptor. For threat value-based defense, the units are threat value per interceptor (where the value scale is entirely arbitrary). For asset value-based defense, the units are asset value per interceptor (where, again, the value scale is arbitrary).

A frequent compromise between threat value and asset value-based defenses that is worthy of mention is a threat value-based defense where the threat value is a function of its expected asset damage independent of other threats. Overlapping damage is assumed small or simply neglected.

Of course, other constraints may be formulated and defense objectives defined in addition to or in lieu of those described above. The predictive model is simply offered as a foundation upon which such can be built.

The Value of Time

One key objective that any many-on-many algorithm must achieve with some level of efficiency is to preserve the number of available shoot-look-shoot engagement opportunities. To accomplish this, the first shot against a threat must be taken as early as possible and subsequent shots must be committed soon after kill assessment for the preceding shot is complete. Obviously, timing constraints and contention for interceptors make this a potentially difficult problem. For example, to achieve the maximum number of shoot-look-shoot opportunities, multiple threats may

contend for the last interceptor at a given launcher. Alternatively, sufficient inventory may be available but launch rate constraints require that shoot-look-shoot opportunities on one or more of the threats be forfeited. It is therefore apparent that the number of shoot-look-shoot opportunities for a given threat cannot be determined absolutely prior to solving the many-on-many problem. In other words, the assignment and scheduling problems are inseparable.

Because MMR is a suboptimal optimization algorithm, the objective function must be designed to compel the algorithm to select from among the earliest engagements and work forwards in time. To induce such "top-down" allocation in the *simplified approach*, the SSKPs must be constrained such that two conditions are met. First, the SSKP for the set of candidate engagements against a given threat should be a function only of the threat type and the intercept time. Second, it should be a non-increasing function of intercept time with the following exception— zero SSKP regions may be superimposed without limit as long as the non-zero portions of the function fulfill the non-increasing requirement.

SSKPs that increase with time represent a difficult problem for any optimization method— how much time can be safely traded for increased SSKP without sacrificing a shoot-look-shoot opportunity? Because such determinations are made for each threat independently, they represent the most optimistic estimates. Therefore, the question persists. If too much time is given up, then a simple launch rate constraint or any other interactive constraint could cost a shoot-look-shoot opportunity. As mentioned earlier, inventory constraints may automatically preclude the expected number of opportunities. Without a more sophisticated optimization method, the best engagement for a given shoot-look-shoot opportunity must be assumed to correspond to the earliest feasible intercept.

Remaining within the confines of MMR for now, representing early engagements as better engagements can be achieved by constraining the SSKP to non-increasing functions of the intercept time as described above. If SSKP is a strictly decreasing function of time, no change to the algorithm as it has been defined is required. However, let us consider here an SSKP dependent only on threat type and therefore constant with respect to time (a common approximation). In this case, the rate of return calculations must be modified to "tip the scales" ever so slightly in favor of earlier engagements. One simple solution is to introduce a new algorithm control parameter which when multiplied by the rate of return expressions represents an "opportunity cost" associated with the advancement of time. In the case of SSKPs dependent only on threat type, the time constant may simply be set very large.

In addition to the cost of lost shoot-look-shoot opportunities, the term may also be assumed to account for the opportunity cost associated with the risk of system saturation as a result of postponing engagements in the face of a growing threat raid size. Note, however, that a potential cost associated with engaging early always exists; interceptors may be expended on low-value engagements that could have been withheld for higher value engagements against subsequently detected threats. However, the minimum rate of return parameter described earlier is the appropriate means by which a balance can be achieved.

Inventory-Limited Defense

In a previous example, the problem of preserving shoot-look-shoot opportunities was complicated by an inventory-limited situation. However, even in the absence of shoot-look-shoot opportunities, inventory-limited defense remains a problem. In effect, local optima are created as a result of finite interceptor resources bounding the solution space. Consequently, *no simple solutions to this problem exist within the confines of simplistic methods such as MMR.*

Responses to the inventory-limited problem fall into four categories. First, use more sophisticated general optimization methods. Current computer processor limitations make the prospects for this approach discouraging. Second, force fit the many-on-many problem to linear methods. Unfortunately, this approach often addresses the inventory-limited problem at the expense of interactive timing and geometry constraints, overlapping asset damage, etc. Third, augment MMR with heuristics specific to addressing inventory-limited situations. Unfortunately, most heuristics are at best only slightly less near-sighted than MMR. Therefore, they cannot consistently recognize the situations they are designed to remedy. As a result, improved inventory-limited performance is achieved at the expense of performance in the absence of inventory limitations. Finally, accept the risks associated with MMR. This author contends that such a response to the problem *is* worthy of consideration. While *gedankenexperiments* that exploit the limitations of MMR can be easily conceived, formulating realistic threat scenarios against realistic defense deployments that can similarly exploit such weaknesses is less trivial. In any case, a reasonable defense deployment will usually mitigate any significant advantage.

Overlapping Asset Damage

A classic problem associated with the application of MMR to engagement planning is that of overlapping damage. Consider the case of two threat objects

attacking a single asset. Each is likely to completely destroy the asset. Because MMR as it has been applied above considers only one additional planned engagement at a time, the rate of return estimate realized by engaging either of the threats individually is zero; the algorithm is incapable of looking far enough ahead to realize the return associated with engaging both. Consequently, the objective function must be augmented to eliminate (i.e., smooth out) local optima. The following approach is suggested.

The total estimated damage on each asset is divided among all of the threats targeting it in proportion to their damage capabilities independent of the other threats. This quantity can be identified as the "linearized damage."

To maximize surviving asset value, a new expression is substituted for the previously defined asset value-based rate of return equation. Using the new form of the rate of return equation and assuming high kill probabilities, half of the asset will appear to be saved if either one of the threats are engaged. If the minimum rate of return is sufficiently low, an engagement will be selected for one of the threats. On the next pass, the nonlinear damage equations will dominate and MMR will determine that the entire asset may be saved if the second threat is engaged. Because MMR normally allocates in order of decreasing rate of return, these two engagements would tend to be selected on consecutive passes since the second engagement would yield a *higher* rate of return than the first engagement. The combination of linear and nonlinear functions will therefore compel the algorithm to complete the engagements necessary to defend an asset once the first engagements have been selected.

Drawdown Strategies

In addition to the primary defense objectives introduced earlier, namely leakage, threat value, and asset value, secondary defense objectives may also exist. Foremost among them is the manner in which interceptors are expended across launchers and launcher sites— drawdown strategies. In effect, the value of an interceptor may vary from launcher to launcher and site to site. Two key objectives in the category of drawdown strategies are balancing the drawdown of interceptors across launcher sites to improve the defense posture against future threats *and* depleting launchers with the lowest inventory to facilitate reloads.

Because secondary objectives tend to be highly subjective, mathematical representations may vary considerably. For this work, several requirements were self-imposed. First of all, the necessary properties of an objective function must be preserved— it must be a single-valued function of the current configuration of the decision variables; it must not depend upon the path taken to arrive at the current configuration. Consequently, it is reversible. Second, the marginal value of the last differential unit of interceptors at a site is unity. In this way, if the control parameters are set such that the interceptor values are constant (i.e., unity), then the objective functions revert to the their original forms.

Consider first the case of balancing drawdown across sites. To reflect this objective in the value of the interceptors, let the interceptor value at each launcher decrease according to a set of user-defined parameters. The relative values of the deflation constant for a specific launcher site (ΔD) define the desired relative balance of interceptors across sites. The values can then be scaled up or down as a set to control the importance of balancing relative to the primary objectives. Very large values essentially disable balancing altogether; small values cause interceptors at inventory-rich sites to be relatively inexpensive from a rate of return standpoint. A critical consideration in establishing the values of these parameters is that the relative importance of balancing may be set to the point that shot opportunities may be sacrificed.

Similarly, the case of depleting launchers with the lowest inventories to enable reloads can be represented. Initially, this will be treated independently of site balancing. In this case, because interceptor values should *increase* with increasing inventory, the user-defined parameter controlling this behavior is labeled an *inflation* constant.

Now consider the problem of combining balancing and depletion factors. This can be accomplished by simply replacing the interceptor inventory at a launcher with the inventory value at a launcher in the site balancing equations.

Similar to the balancing case, large values of the depletion parameter (ΔI) diminish the relative importance of depleting nearly empty launchers; whereas small values enhance its importance, potentially to the point that shoot-look-shoot opportunities may be sacrificed.

With both balancing and depletion cases, the interplay between the time constant introduced earlier and the deflation and inflation constants introduced here will determine the relative influence of these factors. Simple rules of thumb can be formulated based upon the relative spacing of launchers and sites, typical closing speeds, and so forth to determine initial settings for these parameters. As noted previously, the algorithm has no explicit mechanisms for insuring conservation of shoot-look-shoot opportunities. Consequently, the balancing and depletion parameters should be set to insure that these factors are weak

relative to the time constant factor so that these effects occur only at the expense of small amounts of time.

Adaptive Method of Fire

The reader may note that no heuristics, rule sets, or other logic *specific* to achieving particular methods of fire have appeared explicitly in the algorithm design. The question may arise concerning whether or not the algorithm can exploit such strategies as shoot-look-shoot or whether or not it "understands" the basis for reserving salvos for the last shot opportunity. The answer is traceable to the origins of these methods of fire— the statistical merits upon which the strategies were originally founded. These statistical foundations are those that have been represented in the model described above. Because the algorithm is built upon the mathematics and not the discrete strategies derived from them, its inclination to exploit these strategies is inherent. Therefore, while "shoot-look-shoot last-shot salvo" or $(S\text{-}L\text{-})^{m}S^{n}$ is the preferred strategy, the algorithm is not limited to it.

For example, consider a case of two launcher sites, a forward site with 50 interceptors and a rear site with a single interceptor. Consider also a set of 25 inbound threats such that the only available shoot-look-shoot opportunities are those where the first shots are from the forward site and the follow-on shots from the rear site. Assume also a constant SSKP of 0.8, long time constant, weak balancing and depletion, and a rate of return threshold sufficiently low that all interceptors should be expended in any case. Ignoring inventory constraints, the preferred method of fire would be shoot-look-salvo-n. However, inventory constraints dictate shoot-two-look-shoot-one as the optimal method of fire across all the threats ($25*2 = 50$ shots from the forward launcher and $25*[1\text{-}0.8]^{2} = 1$ shot from the rear launcher based upon MVA). Many rule set or heuristic-based algorithms would preclude such seemingly unusual methods of fire; the algorithm described above would attempt to execute such a strategy. In short, attempts to specify or limit methods of fire at any time prior to assignment of weapons are not without risk.

Advantages of Predictive Planning

The advantages of predictive engagement planning are significant. Predictive algorithms tend to be less susceptable to off-nominal situations. They are generally less biased towards certain types of situations or scenarios. Their "first principles" foundation make them receptive to additional features, particularly those of a probabilistic nature. Consider these examples:

- Imperfect Kill Assessment
- Correlated Kill Assessments
- Correlated Kill Probabilities.

Imperfect kill assessment includes both the classification of a kill as a "no kill" and a "no kill" as a kill. Correlated kill assessments include the possibility of a non-lethal hit adversely influencing the kill assessment accuracy of subsequent shots. Correlated kill probabilities may represent one of the most significant and potentially serious obstacles to effective engagement planning. To date, most if not all of the proposed algorithms have assumed that SSKPs are uncorrelated. As a result, these algorithms have an *inherently optimistic predilection*. Factors that may contribute to correlated failures are innumerable— erroneous classification, typing, or discrimination; underestimated track errors; radar biases; underestimated launcher alignment errors; poor uplink data (a particular problem for salvos); and so forth. The correlations may be linked to a threat, radar, launcher, etc. and may vary with time. Temporal correlations in uplink quality represent a factor that, if considered, would tend to further enhance the merits of shoot-look-shoot methods of fire over salvos (assuming a common supporting radar). Predictive many-on-many algorithms have the potential to accurately account for these effects.

A predictive algorithm also possesses advantages with respect to interactive constraints of a geometrical nature such as intercept flashes in the fields of view of other interceptors. Some countermeasures may also fall into this category. The probability of these events occurring is generally high only for the first shot on a threat. Failure to factor in the diminished probability for subsequent shots may lead to excessively cautious planning. The risks inherent in ignoring these events for subsequent shots may be equally serious.

Finally, one of the most attractive advantages of a predictive algorithm is the minimum of operator inputs required for robust behavior over a wide spectrum of situations. The control parameters for the algorithm described above are summarized in the following table:

Symbol	Name	Controls...
R_{min}	Minimum rate of return on investment of resources	Aggressiveness, exchange rate
ΔT	Rate of return decay time constant	Value of time
ΔD	Deflation constant for site m	Site balancing
ΔI	Launcher inflation constant	Launcher depletion for reloads

Software Implementation

Given the potential computational intensity of the many-on-many problem, the manner in which the above model is implemented in software is important. Certain

principles of probability may be exploited to accelerate the computations. The most notable is causality—events can only be influenced by preceding events and can only influence subsequent events. This principle insures no circular dependencies. As candidate engagements are moved in or out of the set of planned engagements, the effects of preceding engagements on the engagement under consideration are evaluated followed by the effects of that engagement on subsequent engagements. Taking advantage of the fact that only one engagement is moved at a time, many sum and product equations can be replaced with forms representing the changes to these quantities, requiring only that the values be appropriately initialized and retained between calculations. Notionally, these may take forms such as

$$Q \leftarrow Q+\Delta \quad (13)$$
$$Q \leftarrow Q-\Delta \quad (14)$$
$$Q \leftarrow Q*\Delta \quad (15)$$
$$Q \leftarrow Q/\Delta \quad (16)$$

The Integrated Approach

One goal of the simplified approach was to assess the quality of the basic predictive many-on-many algorithm somewhat independently from processing and memory considerations. In the simplified form, a relatively exhaustive set of candidate engagements is calculated first, in effect, mapping the entire solution space for a single threat independent of other threats. This set of engagements is then passed to the many-on-many algorithm. This is the method currently represented in TISES; a more computationally efficient approach (the *integrated approach*, as it shall be called) is described below.

The processes of generating and selecting from the set of candidate engagements account for the bulk of the engagement planning computations. The computational burden of both processes scales with the number of candidate engagements. It is therefore apparent that if candidate engagements can be generated selectively, the time required for both processes can be reduced substantially. However, selective generation requires integration of the generation and selection processes. In effect, as the selection process continues, searching through the solution space as defined by the candidate engagements, additional candidates are added representing regions of the solution space adjacent to those already explored. The solution space is mapped on an as-needed basis. The result is not unlike branch-and-bound techniques.

The net effect of this integrated approach is that fewer engagements are generated *and* fewer are considered by the many-on-many process. A third advantage relevant to the real-time environment is that if time expires for the engagement planning process, it is more likely to exit with at least some desirable engagements selected for execution. A fourth advantage is that first commits, salvos, and shoot-look-shoot shots can be more efficiently packed in time. In the simplified approach, the engagement timing precision is limited by the time granularity of the candidate engagement set, typically on the order of the planning cycle time.

Integration of the generation and selection processes requires the creation of some additional supporting algorithms. Most significant is an algorithm that can generate an engagement for a given threat and launcher with the earliest intercept time corresponding to a commit time greater than and, assuming non-interactive constraints are not violated, within some tolerance of a specified time. The specified time may be the planning end time, the commit time corresponding to the launch time of another engagement (plus some launch spacing if the same launcher), or the kill assessment time of another engagement (plus some delay associated with the planning cycle). These three correspond to earliest engagements, salvo engagements, and shoot-look-shoot engagements. If non-interactive constraints introduce a "hole" in the battlespace, then the earliest feasible intercept within some tolerance after the hole is returned. If no feasible engagements exist, then no engagement is returned and no error condition is required. Because launch time may be a very complex function of intercept time for a given threat/launcher combination, this algorithm is non-trivial. The algorithm must also maintain records of battlespace holes (by threat, launcher, radar, etc., as required) to prevent redundant traverses of infeasible regions.

The integrated approach consists of three main parts organized as follows:

1) Generate initial set of candidate engagements
Loop until rate of return threshold can no longer be met:
  2) Find candidate engagement with best rate of return
  3) Add best engagement to planned engagements and generate additional candidates as needed
End loop

The algorithm is shown in greater detail in Figure 1. Italics indicate those parts of the algorithm unique to the integrated approach.

Because launcher inventory is treated as a continuous quantity and is never allowed to be negative, "non-empty" in the context of the algorithm may best be defined as greater than some small positive number in order to prevent futile attempts at expending every last "fraction" of an interceptor from a given launcher.

```
1) Generate initial set of candidate engagements:
loop over Threats
  Generate Candidate Engagement (Earliest Intercept)
    from each non-empty launcher
  loop over Existing Engagements on this Threat
    Generate Candidate Engagement (Next SLS) from
    each non-empty launcher
  end loop
end loop

loop -- main
  2) Find candidate engagement with best rate of
    return:
  Best rate of return = - large number
  loop over Candidate Engagements
    Move to Planned Engagements (temporarily) and
      determine rate of return
    if Launcher Inventory ≥ 0.0 and Rate of return ≥
      maximum (Best rate of return, rate of return
      threshold) then
      Mark Candidate Engagement as incumbent best
    end if
    Move back to Candidate Engagements
  end loop
  3) Add best engagement to planned engagements and
    generate additional candidates:
  if Best rate of return ≥ rate of return threshold then
    -- a qualifying engagement is found
    if Interactive Timing and Geometry Constraints
      violated then
      Generate Candidate Engagement (Next Salvo)
        from this launcher if not empty
      Delete best engagement
    else
      Move best engagement to Planned Engagements
        (permanently)
      Generate Candidate Engagement (Next Salvo)
        from each non-empty launcher
      Generate Candidate Engagement (Next SLS) from
        each non-empty launcher
    end if
  else
    exit main loop
  end if
end loop -- main
```

Figure 1. The Integrated Algorithm.

Acknowledgments

The TISES effort for which the algorithm described herein was developed is currently sponsored by the THAAD Project Office System Engineering Division under the auspices of the THAAD Software Independent Validation and Verification contract number DASG60-94-C-0016. Past sponsorship has been provided by the Ground Based Element System Simulation Development contract number DASG60-92-C-0173. The author wishes to thank Mr. Tony Cosby, Mr. Doug Engle, Ms. Dawn Horn, and Mr. Rich Mullin of TPO System Engineering Division for their support of this work. The author also wishes to thank Phil Colvert and Gary Makowski of Tec-Masters, Incorporated and Daniel Dean and Anthony Glover of Coleman Research Corporation for their valuable suggestions and thorough critique of this paper.

AUTHOR INDEX

Advani, S., 12 (11-MST-1), WD (11-MST-1), 458 (73-MST-16)
Albronda, J., 12 (11-MST-1)
Allen, R., 83 (13-MST-3)
Annos, J., NA (73-MST-16)
Asburn, J., NA (14-MST-4)
Baker, M., 213 (43-MST-9)
Beavin, W., 131 (27-MST-5)
Belanger, F., 204 (43-MST-9)
Bell, J., 481 (56-MST-11), 490 (71-MST-14)
Bennett, P., 387 (71-MST-14)
Bettendorf, S., 458 (73-MST-16)
Bezdek, W., 376 (71-MST-14)
Biezad, D., NA (70-MST-13)
Bleak, J., NA (28-MST-6)
Bonner, M., NA (42-MST-8)
Bouchard, C., 90 (14-MST-4)
Bouwens, C., 139 (27-MST-5)
Broussard, J., 90 (14-MST-4)
Brown, M., 431 (72-MST-15)
Bucco, D., 161 (28-MST-6)
Buffington, J., 305 (70-MST-13)
Byrd III, L., NA (14-MST-4)
Cameron, D., NA (70-MST-13)
Cardullo, F., 23 (11-MST-1), 412 (72-MST-15)
Charbonneau, M., 62 (13-MST-3)
Chisholm, J., 161 (28-MST-6)
Christman, J., 224 (43-MST-9)
Chubb, G., 69 (13-MST-3), 75 (13-MST-3)
Chung, J., 52 (11-MST-1)
Chung, W., 1 (11-MST-1)
Colgren, R., 275 (70-MST-13)
Colvert, P., 121 (14-MST-4)
Conzachi, K., 359 (71-MST-14)
Cord, T., 259 (56-MST-11), 285 (70-MST-13)
Cyril, X., 204 (43-MST-9)
Dahms, M., 316 (70-MST-13)
De Jonckheere, R., 194 (43-MST-9)
Debitetto, P., 41 (11-MST-1)
Devilbiss, J., 154 (28-MST-6)
Dingman, B., 213 (43-MST-9)
Egger, W., NA (11-MST-1)
Enns, D., 275 (70-MST-13)
Exter, J., 397 (71-MST-14)
Eynden, H., WD (11-MST-1)
Flitter, L., NA (11-MST-1)
Forrest, D., 224 (43-MST-9)
Franklin, T., 194 (43-MST-9)
Fukuda, Y., 471 (73-MST-16)
Fullford, D., 147 (27-MST-5)
Funk, J., WD (11-MST-1)
Gaublomme, D., WD (11-MST-1)
Gemmill, C., 90 (14-MST-4)
George, G., 412 (72-MST-15)
Ghosh, T., 231 (43-MST-9)
Gingras, D., NA (42-MST-8)
Gregg, W., 213 (43-MST-9)
Grismer, D., 305 (70-MST-13)
Haas, D., NA (11-MST-1)
Haddix, F., WD (27-MST-5)
Haeck, N., 12 (11-MST-1)
Haines, R., 181 (42-MST-8)
Halley, T., 376 (71-MST-14)
Hammond, J., WD (27-MST-5)
Harbaugh, G., NA (14-MST-4)
Hartman, R., 90 (14-MST-4)
Heaton III, H., 154 (28-MST-6)
Hensley, D., 296 (70-MST-13)
Hiwada, T., 471 (73-MST-16)
Hogue, J., 83 (13-MST-3)
Horn, D., 121 (14-MST-4), NA (14-MST-4)
Hosman, R., 402 (72-MST-15)
Howells, P., 62 (13-MST-3)
Hummel, P., 376 (71-MST-14)
Hunter, R., 69 (13-MST-3)
Jategaonkar, R., 248 (56-MST-11)
Johnson, E., 41 (11-MST-1)
Kageyama, K., 471 (73-MST-16)
Kaplan, J., 366 (71-MST-14)
Kenney, P., 366 (71-MST-14)
Kerr, B., 467 (73-MST-16)
Kim, T., 190 (42-MST-8)
Kindel, W., 316 (70-MST-13)
Kircher Jr., R., 351 (71-MST-14)
Kwan, B., 62 (13-MST-3)
Lash, S., 481 (56-MST-11)
Lee, K., 422 (72-MST-15)
Leite, S., 90 (14-MST-4)
Liebich, R., 316 (70-MST-13)
Lin, Y.-Y., 240 (43-MST-9)
Liu, S.-S., 240 (43-MST-9)
Lubbers, S., WD (27-MST-5)
Macy, P., 75 (13-MST-3)
McCabe, M., NA (13-MST-3)
McCool, K., NA (11-MST-1)
McCrillis, J., WD (11-MST-1)
McGlamery, B., 194 (43-MST-9)
McKeehen, P., 285 (70-MST-13), 305 (70-MST-13)
Mergens, D., 194 (43-MST-9)
Miro, J., NA (43-MST-9)
Mönnich, W., 248 (56-MST-11)
Moyer, S., 110 (14-MST-4)
Myatt, J., 305 (70-MST-13)
Nahon, M., 12 (11-MST-1)
Nalepka, J., 100 (14-MST-4)
Neville, K., 172 (42-MST-8)
Nguyen, B., 259 (56-MST-11)
Nichols, J., WD (11-MST-1)
O'Rourke, M., 481 (56-MST-11), 490 (71-MST-14)
Paulson, C., 333 (70-MST-13)
Persing, T., WD (27-MST-5)
Ralston, J., 481 (56-MST-11)
Robinson, D., 1 (11-MST-1)
Rodriquez, R., 224 (43-MST-9)
Romanoff, C., 30 (11-MST-1)
Roumeliotis, L., 449 (73-MST-16)
Rushenko, A., 326 (70-MST-13)
Ryabkov, V., 326 (70-MST-13)
Sadovnitchii, S., 326 (70-MST-13)
Sanderson, J., NA (13-MST-3)
Sandoval, J., 326 (70-MST-13)
Sanford, B., 422 (72-MST-15)
Sato, H., 471 (73-MST-16)
Schmidt, M., 333 (70-MST-13)
Schmidt, O., 316 (70-MST-13)
Schroeder, J., 1 (11-MST-1)
Scribner, K., WD (27-MST-5)
Scrip, B., 121 (14-MST-4), NA (14-MST-4)
Shima, K., 471 (73-MST-16)
Shin, S., 269 (56-MST-11)
Shiomi, K., 471 (73-MST-16)
Slater, M., 431 (72-MST-15)
Slattery, R., 422 (72-MST-15)
Smith, R., 83 (13-MST-3)
Stadler, B., 100 (14-MST-4)
Stegner, J., NA (11-MST-1)
Sullivan, P., 52 (11-MST-1)
Thompson, R., NA (14-MST-4)
Thompson, S., WD (27-MST-5)
Valentino, G., WD (27-MST-5)
Van Zuylen, E., 458 (73-MST-16)
Verbeek, R., WD (11-MST-1)
Walker, W., 351 (71-MST-14)
Watkinson, K., 110 (14-MST-4)
Werme, E., 90 (14-MST-4)
Wiederholt, M., 224 (43-MST-9)
Williams, G., 341 (71-MST-14)
Wright., D., 90 (14-MST-4)
Wu, W., 23 (11-MST-1)
Yoon, S., 269 (56-MST-11)
Zabek, A., 139 (27-MST-5)
Zammit, S., 442 (73-MST-16)



For permission to copy or republish, contact:
American Institute of Aeronautics and Astronautics
1801 Alexander Bell Drive
Suite 500
Reston, VA 20191-4344

ISBN
1-56347-236-8